1972

Computer-aided circuit optimization including the adjoint method of calculating network sensitivities

John Walter Olson

Department:

111

COMPUTER-AIDED CIRCUIT OPTIMIZATION INCLUDING THE ADJOINT METHOD OF CALCULATING NETWORK SENSITIVITIES

BY

JOHN WALTER OLSON, 1949-

A THESIS

Presented to the Faculty of the Graduate School of the

UNIVERSITY OF MISSOURI - ROLLA

In Partial Fulfillment of the Requirements for the Degree

MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

1972

Approved by

Robert C. Pearson (Advisor) Paul D. Stigall

James K. Byers

ABSTRACT

A complete discussion of the optimization of a circuit, including modeling and programming difficulties encountered, is presented. A three transistor bandpass filter is used in order to study various methods of optimization. The adjoint method of calculating the network error gradient vector is compared to the perturbation method. The adjoint method proves to have decided advantages over the other method. A generalized computer program is presented to carry out the Fletcher-Powell optimization algorithm with specialized subroutines which include a network equation solver and both methods of finding the gradient vector. The program proved very effective in realizing a desired transfer function for the given circuit.

ACKNOWLEDGEMENT

The author would like to express his sincere appreciation to Dr. Robert C. Peirson for suggesting this topic and for his assistance during this project.

Thanks are also due Sandra Davis for her help in typing this paper.

TABLE OF CONTENTS

Table of Contents (continued) Page

LIST OF ILLUSTRATIONS

LIST OF TABLES

NOMENCLATURE

C_i	capacitive element
^oC	degrees centigrade
E	adjoint method error functional
F	error functional
h_{fe}	common emitter forward current transfer ratio
h_{ie}	common emitter input impedance
h_{oe}	common emitter output admittance
h_{re}	common emitter reverse voltage transfer ratio
$\underline{H}$	Fletcher-Powell approximation to the Hessian matrix
I_I	port excitation phasor, current source
I_V	actual port response phasor, current
$\hat{I}_V$	desired port response phasor, current
n_I	number of independent current sources
n_V	number of independent voltage sources
Q	Q factor defined in reference 7
R_i	resistive element
V	actual output voltage phasor
$\hat{V}$	desired output voltage phasor
V_I	actual port response phasor, voltage
V_V	port excitation phasor, voltage source
$\hat{V}_I$	desired port response phasor, voltage
W_I	weighting function for current sources
W_V	weighting function for voltage sources
$\underline{x}$	vector containing circuit elements
α	scalar variable used in Fletcher-Powell technique

γ	adjoint multiplier used in calculating gradient
ε	specified scalar error
Π	given electronic network
$\overline{\Pi}$	adjoint network
$\sum_{x}$	the summation over x
ϕ_B	B branch adjoint current phasor
ψ_B	B branch adjoint voltage phasor
ω	radian frequency
Ω	specified frequency range
$\frac{\partial}{\partial}$	partial derivative with respect to x
$\underline{\nabla F}$	error functional gradient vector
Δx	delta x or the change in x
Re{}	the real part of a complex argument
()*	the conjugate of a complex argument

I. INTRODUCTION

There have been many papers written on different methods used in optimizing a given circuit.[1,4,5,9,10] Most of these discussions stress the derivation of the method and possibly list some results. Rarely is there a complete discussion, from start to finish, of optimizing a circuit, including modeling and programming difficulties encountered. This paper is intended to partially fill the need for more information of a practical nature on optimization. The goal is not to perform a detailed and comprehensive analysis of any one circuit but to show the use of several optimization methods. In addition the superiority of the recently developed adjoint network technique of calculating network sensitivities is demonstrated.[1,2,3]

After choosing a circuit, it was constructed in the laboratory. Circuit response data were taken in order to develop a reasonable mathematical model which could be used in a computer program and compared with the actual circuit response. Optimization techniques using this model were applied, and the optimized values of the circuit elements used in the original circuit. In this manner, the effectiveness of the optimization techniques used, the relative ease of programming, the amount of computer time used, and any difficulties encountered

could be studied. The Fletcher-Powell optimization technique[9] was chosen because of its ease of programming and its approximate quadratic convergence. This technique requires the calculation of a series of element sensitivities which make up a network gradient vector. The usual method of computing the element sensitivities is by perturbation. However, this method uses a relatively large amount of computer time because a series of complete network analyses must be made over the frequency range of interest for every element in the circuit. Another recently developed technique to obtain the element sensitivities is the adjoint method. This method has the advantage of requiring only two network analyses and therefore results in a considerable savings of computer time. Both methods were used in order to compare their effectiveness.

II. REVIEW OF THE LITERATURE

Prior to a detailed discussion of optimization techniques and the adjoint network method, a brief summary of the pertinent literature is in order.

Temes and Calahan[4] give a general discussion covering the state of the art of optimization methods. Calahan[5] goes into the specific details of various techniques. Dawson, Kuo, and Magnuson[13] compare several different pre-packaged network analysis programs.

Director and Rohrer[1,2] have thoroughly outlined the derivation and application of the adjoint network method in the time and frequency domains and further generalize the method to include nonlinear elements.[3]

III. OPTIMIZATION

A. The Problem

In optimizing any circuit the problem is one of adjusting or aligning a network which fails to meet a prescribed performance criterion. The criterion chosen in this case was that of realizing a given frequency response.

The procedure used will be concerned with the minimization of a single function,

$$F(\underline{x},\omega),$$

where $\underline{x}$ is a vector made up of the circuit elements and ω is radian frequency. This error functional was chosen to be one half the integral of the square of the difference between the desired output voltage, $\hat{V}(\omega)$, and the actual output voltage, $V(\omega)$,

$$F(\underline{x},\omega) = \frac{1}{2} \int [\hat{V}_j(\omega) - V_j(\omega)]^2 \, d\omega \, .$$

If F is a well-behaved function of $\underline{x}$ and the only knowledge of F at each iteration is its gradient, then there is no better procedure for minimization than a motion in the direction of the negative gradient.[5]

It must be remembered that there exists no method at present, short of a complete and time wasting search of the error functional surface, for the determination of an overall, or global, minimum. The best that can be hoped for with any method is convergence to a local minimum.

B. The Fletcher-Powell Optimization Technique

A method of optimization which quickly finds the minimum of a general quadratic equation must be used in order to guarantee rapid convergence. This is due to the dominance of the second-order terms of a Taylor series expansion near the minimum.[9]

The Fletcher-Powell minimization procedure uses knowledge of F and the error functional gradient vectors $\underline{\nabla F}$, at previous iterations to improve the current iteration. The method generates a vector, $\underline{s}^j$, at each iteration, which is orthogonal to all previous vectors, $\underline{s}^i$, i=1,2, . . . q, and one which gives a direction of decreasing F. The actual method is as follows:

Let $\underline{H}^j$ be any positive definite matrix. At the j-th iteration define

$$\underline{s}^j = -\underline{H}^j \nabla F(x^j)$$

and

$$\underline{x}^{j+1} = \underline{x}^j + \alpha^j \underline{s}^j .$$

Then α^j is chosen to minimize

$$F(\underline{x}^j + \alpha^j \underline{s}^j).$$

This value of α^j is $\hat{\alpha}^j$. Define

$$\underline{\sigma}^j = \hat{\alpha}^j \underline{s}^j ,$$

$$\underline{d}^j = \underline{\nabla F}(\underline{x}^{j+1}) - \underline{\nabla F}(\underline{x}^j) ,$$

$$A^j = \frac{\underline{\sigma}^j \, \underline{\sigma}^{j^T}}{\underline{\sigma}^{j^T} \, \underline{d}^j} ,$$

$$\underline{B}^j = \frac{\underline{H}^j \underline{d}^j \underline{d}^{j^T} \underline{H}^j}{\underline{d}^{j^T} \underline{H}^j \underline{d}^j},$$

and

$$\underline{H}^{j+1} = \underline{H}^j + \underline{A}^j - \underline{B}^j.$$

It can be shown that this selection of $\underline{x}^{j+1}$ and $\underline{H}^{j+1}$ decreases F.[9] This procedure is carried out until two successive iterations produce values of F within a specified error, ε, of each other.

C. Calculating the Network Gradient Vector

From the previous discussion of optimization it is apparent that the network gradient vector, ∇F, made up of the partial derivatives of F with respect to the x^j, must be calculated at each iteration. When no exact mathematical expression for the network error functional can be derived, these partials may be approximated using the method of perturbation,

$$\frac{\partial F}{\partial x_i} = \frac{F(x_1, x_2, \ldots, x_i + \Delta x, \ldots, \omega) - F(\underline{x}_i)}{\Delta x}.$$

This method requires a complete circuit analysis over the frequency spectrum of interest for each element! Thus, an exceedingly large amount of computer time must be used.

D. The Adjoint Method

Recently a method of calculating the network sensitivities with only two network analyses has been studied.[1,2,3]

This is the adjoint network method. Since this is a relatively new method a more detailed discussion of its derivation and implementation will be presented.

Given a linear, fixed structure, lumped, time-invariant network, Π, composed of resistances, inductances, capacitances, transformers, gyrators, and voltage and current controlled voltage and current sources, the problem is to determine what element values, $\underline{x}$, will yield desired responses to given excitations. The desired current responses will be currents through independent port voltage sources and the desired voltage responses will be voltages across independent port current sources.

The port excitation phasors are given by

$$V_{V_i} \qquad i = 1,\ldots, n_V$$

and

$$I_{I_j} \qquad j = 1,\ldots, n_I.$$

These yield the desired port response phasors

$$\hat{V}_{I_j} \qquad j = 1,\ldots, n_I$$

and

$$\hat{I}_{V_i} \qquad i = 1,\ldots,n_V,$$

where n_V is the number of independent voltage sources and n_I is the number of independent current sources.

The error criteria will be a weighted integral squared error over the specified frequency range, Ω.

The advantage of this type of error criteria will be apparent later.

$$E = \sum_{V} \int_{\Omega} \frac{1}{2} W_V(\omega) \left| I_V(\omega) - \hat{I}_V(\omega) \right|^2 d\omega + \sum_{I} \int_{\Omega} \frac{1}{2} W_I(\omega) \left| V_I(\omega) - \hat{V}_I(\omega) \right|^2 d\omega \tag{1}$$

where I_V and V_I are actual port response phasors and W_V and W_I are real, non-negative weighting functions. The weighting functions allow the designer to emphasize certain portions of the frequency response and also allow for an excitation at a given port with no accompanying response.

The partial derivatives of E with respect to the element parameters $\underline{x}$ are needed. Taking the partial of (1) with respect to x_i yields

$$\frac{\partial E}{\partial x_i} = \sum_{V} \int_{\Omega} \mathrm{Re}\{W_V(\omega) [I_V^*(\omega) - \hat{I}_V^*(\omega)] \frac{\partial I_V(\omega)}{\partial x_i}\} d\omega + \sum_{I} \int_{\Omega} \mathrm{Re}\{W_I(\omega) [V_I^*(\omega) - \hat{V}_I^*(\omega)] \frac{\partial V_I(\omega)}{\partial x_i}\} d\omega \tag{2}$$

so $\dfrac{\partial I_V(\omega)}{\partial x_i}$ and $\dfrac{\partial V_I(\omega)}{\partial x_i}$

must be found.

Assume a topologically equivalent adjoint network, of the original network. Let $\Psi_B(\omega)$ and $\Phi_B(\omega)$ be the B-branch voltage and current phasors respectively, in $\overline{\Pi}$. Tellegen's Theorem[7] states

$$\sum_B V_B(\omega)\ \Phi_B(\omega) = 0$$

and (3)

$$\sum_B I_B(\omega)\ \Psi_B(\omega) = 0.$$

During the design procedure V_B and I_B will be changed to $V_B + \Delta V_B$ and $I_B + \Delta I_B$ as the elements change from x to $x + \Delta x$ but since the network topology is invariant, Tellegen's Theorem gives

$$\sum_B [V_B(\omega) + \Delta V_B(\omega)]\Phi_B(\omega) = 0$$

and (4)

$$\sum_B [I_B(\omega) + \Delta I_B(\omega)]\Psi_B(\omega) = 0.$$

Subtracting (3) from (4) yields

$$\sum_B \Delta V_B(\omega)\Phi_B(\omega) = 0 \quad (5)$$

and

$$\sum_B \Delta I_B(\omega)\ \Psi_B(\omega) = 0, \quad (6)$$

and subtracting (6) from (5) gives

$$\sum_B [\Delta V_B(\omega)\Phi_B(\omega) = \Delta I_B(\omega)\Psi_B(\omega)] = 0. \quad (7)$$

In order to find

$$\frac{\partial V_I}{\partial x} \quad \text{and} \quad \frac{\partial I_V}{\partial x}$$

as required in (2) an expression that depends only upon parameter and source variations is needed. In order to find this expression (7) must be made independent of all ΔV and ΔI terms in non-source branches. This is accomplished by considering the resistive, capacitive, inductive, and other types of branches individually. For each non source branch, B, an equation can be derived which is independent of ΔV_B and ΔI_B terms and can be written in terms of some γ_B times the branch parameter variation, ΔX_B. Detailed derivations of these relationships are given in Reference 1.

Once all the adjoint relations are written in a manner so as to make (7) solely dependent upon the $\underline{\Delta x}$ terms, (7) can be written as

$$\sum_V [\Delta V_V \Phi_V - \Delta I_V \Psi_V] + \sum_I [\Delta V_I \Phi_I - \Delta I_I \Psi_I] = \underline{\gamma}^T \underline{\Delta x} \qquad (8)$$

where $\underline{\gamma}$ and $\underline{\Delta x}$ are given in Table I.

If the independent voltage and current sources are held constant so that

$$\Delta V_V = 0$$

and

$$\Delta I_I = 0,$$

(8) becomes

$$\sum_I \Delta V_I \Phi_I - \sum_V \Delta I_V \Psi_V = \underline{\gamma}^T \underline{\Delta x} \; . \qquad (9)$$

Let

$$\Psi_V(\omega) = W_V(\omega) [I_V^*(\omega) - \hat{I}_V^*(\omega)] \qquad (10)$$

be the adjoint network independent voltage sources and

$$\Phi_I(\omega) = W_I(\omega) [V_I^*(\omega) - \hat{V}_I^*(\omega)] \qquad (11)$$

be the adjoint network independent current sources.

TABLE I

ADJOINT RELATIONSHIPS FOR VARIOUS CIRCUIT ELEMENTS

Element Type	Branch Relation in Adjoint	Sensitivity (Component of γ)	Component of Δx
Resistor	$\Psi_R = R\Phi_R$	$-I_R\Phi_R$	ΔR
Capacitor	$\Phi_C = j\omega C\Psi_C$	$j\omega VC\Psi_C$	ΔC
Inductor	$\Psi_L = j\omega L\Phi_L$	$-j\omega I_L\Phi_L$	ΔL
Current Controlled Current Source	$\Psi_{ICI} = -\beta\Psi_{IDI}$ $\Psi_{IDI} = 0$	$I_{ICI}\Psi_{IDI}$	$\Delta\beta$

Substituting (10) and (11) into (9) gives

$$\sum_{V} \{W_V(\omega)[I_V^*(\omega) - \hat{I}_V^*(\omega)] \frac{\Delta V_I}{\Delta x}\}$$

$$+ \sum_{I} \{W_I(\omega)[V_I^*(\omega) - \hat{V}_I^*(\omega)] \frac{\Delta I_V}{\Delta x} = \underline{\gamma} .$$

Taking the real part of this equation and integrating over Ω we see that this equation is equivalent to (2), thus

$$\frac{\partial E}{\partial x} = \int_{\Omega} \mathrm{Re}\{\gamma\} \, d\omega.$$

The network gradient vector may, therefore, be obtained by performing one analysis of the original circuit and one analysis of the adjoint network over the frequency range of interest in order to obtain γ.

IV. THE CIRCUIT

A. Initial Circuit

Figure 1 shows the active bandpass filter chosen for optimization. The circuit was originally designed for use in filtering three pilot tones out of seven in a wireless communication system. To call one of twenty-five special receivers at different locations a combination of three pilot tones filtered out of seven can be used to switch the receivers on. These filters require a high Q factor.[8] The circuit has a high thermal stability up to 80° C. due to the dc feedback made possible by the zener diode.

The passive twin T notch filter[14] consisting of elements R7, R8, R9, C3, C4, and C5 was built using matched components. A relatively high gain was obtained permitting small input signals and allowing a high Q and an output voltage of several volts. With a maximum input of 100 millivolts, the maximum output is 1.72 volts. The center frequency is 5.3087 kHz and the total dc current drain is 4 milliamps when a 24 volt supply is used.

The circuit was constructed and data for frequency response was tkaen. Results are shown in Table II and a plot of gain in decibels vs. log frequency in shown in Figure 2.

The common emitter h parameter equivalent circuit was chosen to model the three transistors because it

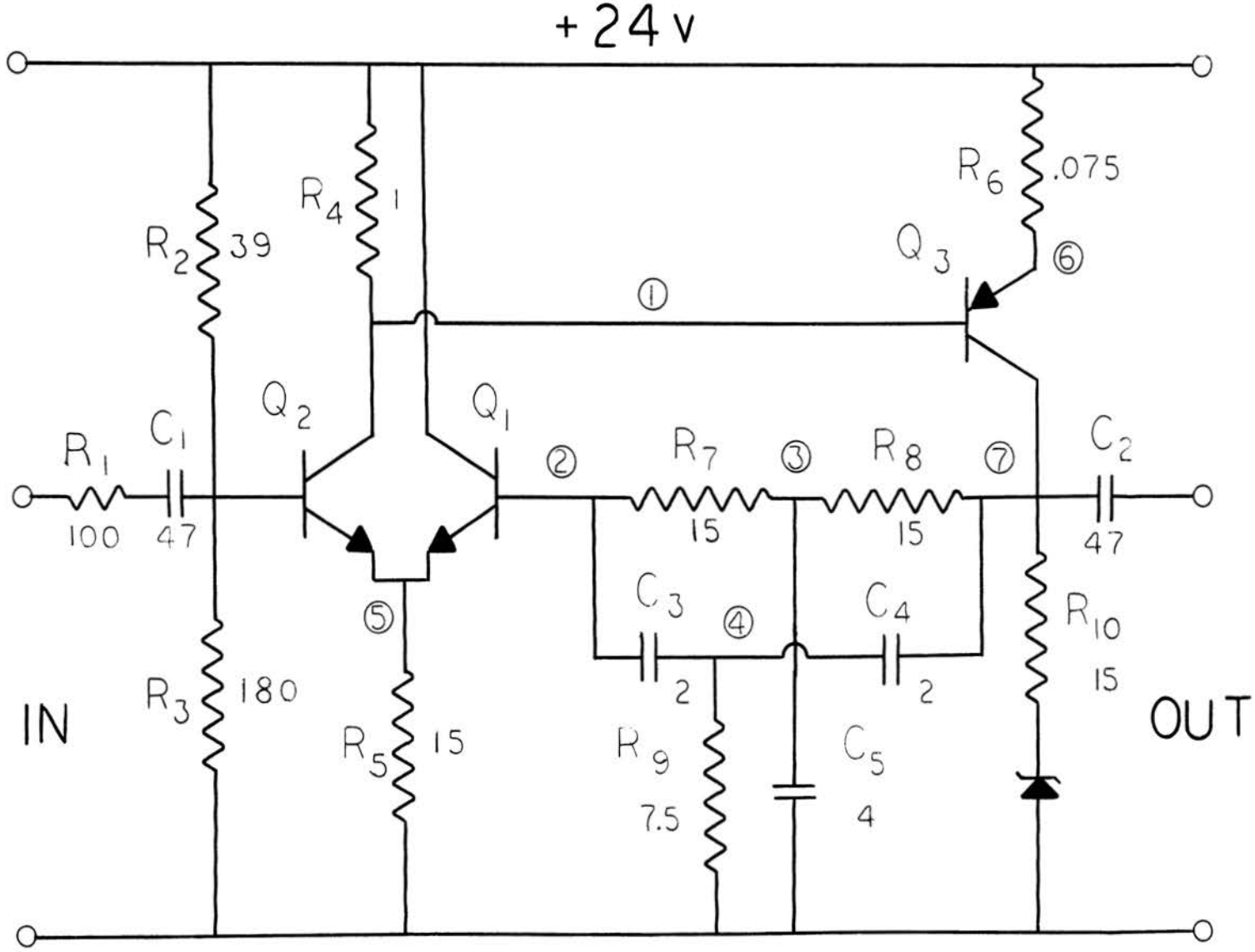

Figure 1. Narrow Band High Q Filter Chosen For Optimization

TABLE II

OUTPUT VOLTAGE AND GAIN VS. FREQUENCY
FOR THE ORIGINAL CIRCUIT

Frequency (kHz)	V_{out} (mV)[1]	Gain (db)
.1	11.0	-13.2
.2	11.0	-13.2
.4	12.0	-12.4
.6	13.0	-11.7
1.0	14.8	-10.6
2.0	23.6	-6.7
4.0	92.3	4.3
5.0	413.0	18.4
5.308	860.0	24.2
6.0	164.0	10.3
10.0	35.0	-3.1
20.0	16.0	-9.9
40.0	12.7	-12.0
60.0	11.2	-13.0
100.0	9.9	-14.1

[1]Data taken with a sinewave input of 50 mVrms. R_6 was set at 75Ω.

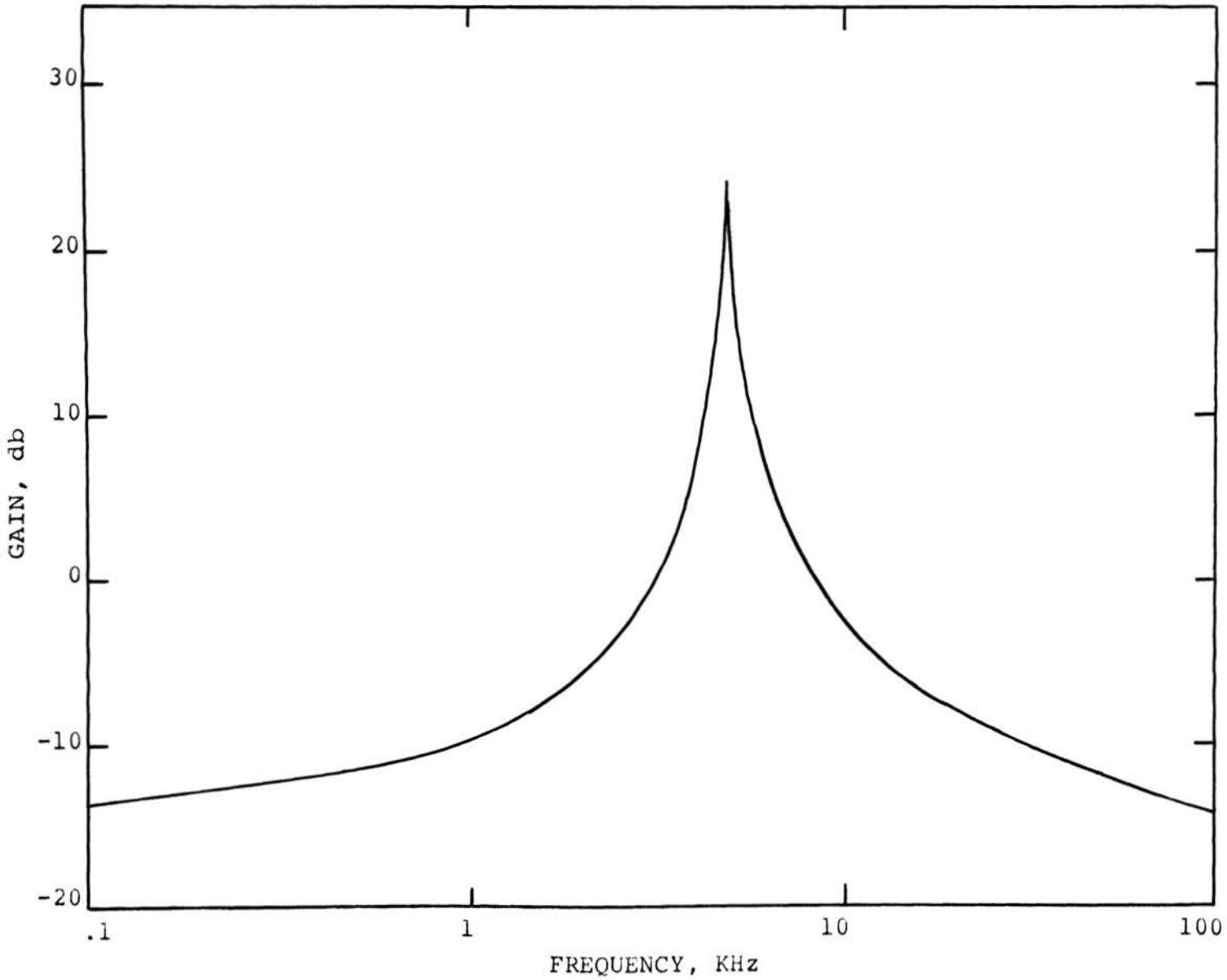

Figure 2. Gain vs. Frequency For Bandpass Filter

ideally fits the low input and high output impedances of the transistor and at audio frequencies it was as complete a model as necessary. The h parameters were also easily obtainable using a curve tracer. The measured h parameters used in the final model are given in Table III.

TABLE III

MEASURED h PARAMETERS FOR THE THREE GIVEN TRANSISTORS

Transistor	h_{fe}	h_{ie} (ohms)	h_{oe} (mhos)	h_{re}
Q1	260	835	$4x10^{-5}$	0
Q2	240	1000	$4x10^{-5}$	0
Q3	190	500	$4x10^{-5}$	0

B. The Circuit Model

Using the h parameter model for the three transistors, the audio frequency equivalent circuit of Figure 3 is obtained. The zener diode was omitted in the circuit model since its measured dynamic resistance was much less than R10. In the original circuit, elements C1, C2, and R1 are used to couple the input and the output signals. The capacitors, because of their relatively large values, were omitted in the circuit model in order to simplify it. This omission does not affect the optimization, however.

A seven by seven nodal admittance matrix was obtained from the node voltage equations of the network as shown in Figure 4. This matrix was verified by using the Electronic

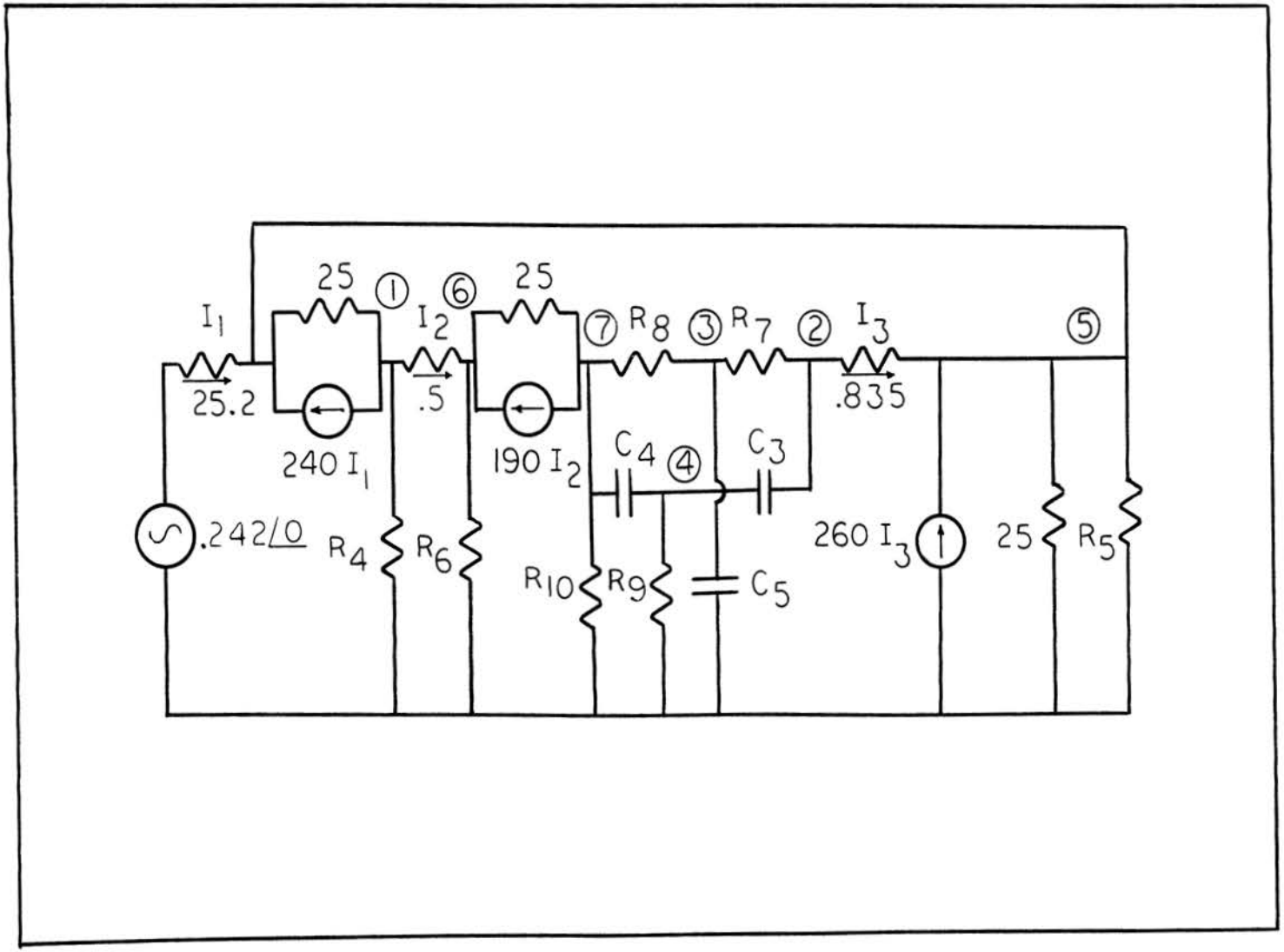

Figure 3. Audio Frequency Equivalent Circuit For Bandpass Filter

$$
\begin{bmatrix}
2.04+G_4 & 0 & 0 & 0 & 9.56 & -2 & 0 \\
0 & 1.2+G_6+j\omega C_3 & -G_7 & -j\omega C_3 & -1.2 & 0 & 0 \\
0 & -G_7 & G_7+G_8+j\omega C_5 & 0 & 0 & 0 & -G_8 \\
0 & -j\omega C_3 & 0 & G_9+j\omega(C_3+C_4) & 0 & 0 & -j\omega C_9 \\
-.04 & -312 & 0 & 0 & 321.55 & 0 & 0 \\
-382 & 0 & 0 & 0 & 0 & 382.04+G_6 & -.04 \\
380 & 0 & -G_8 & -j\omega C & 0 & -390.04 & .04+G_8+G_{10}j\omega C_4
\end{bmatrix}
\begin{bmatrix} V_1 \\ V_2 \\ V_3 \\ V_4 \\ V_5 \\ V_6 \\ V_7 \end{bmatrix}
=
\begin{bmatrix} -2.3 \\ 0 \\ 0 \\ 0 \\ -2.31 \\ 0 \\ 0 \end{bmatrix}
$$

Figure 4. Nodal Admittance Matrix for Original Circuit Model

Circuit Analysis Program (ECAP).[11] This matrix must be inverted to solve for the branch voltages and currents. The nodal admittance matrix was used because, unlike the impedance matrix, it is relatively sparse.[6] Since it is sparse, a standard matrix inversion subroutine would use up computer time with needless operations. For this reason and since the network equations must be solved many times in any optimization procedure, a sparse matrix technique was employed. The matrix reduction technique used takes advantage of the zero terms of the matrix by using the technique explained by Sato and Tinney[6] of ordering the network equations so that a near minimal amount of computer time and storage is used. Further work has been done in this area by Tinney and Walker.[12]

Each step of the matrix inversion was written separately in the final program, omitting all operations that would involve zero elements. Once the circuit analysis subroutine, called NETWK, was written it was compared with a standard matrix inversion subroutine[5] (Appendix A). The standard subroutine took approximately one minute to completely solve the network at one hundred different frequencies; the specialized program NETWK, gave the same results in only seventeen seconds of computer time--a considerable savings in the final optimization program.

The voltage gain of the model vs. log frequency is shown in Figure 5. It can be seen that the model is a close

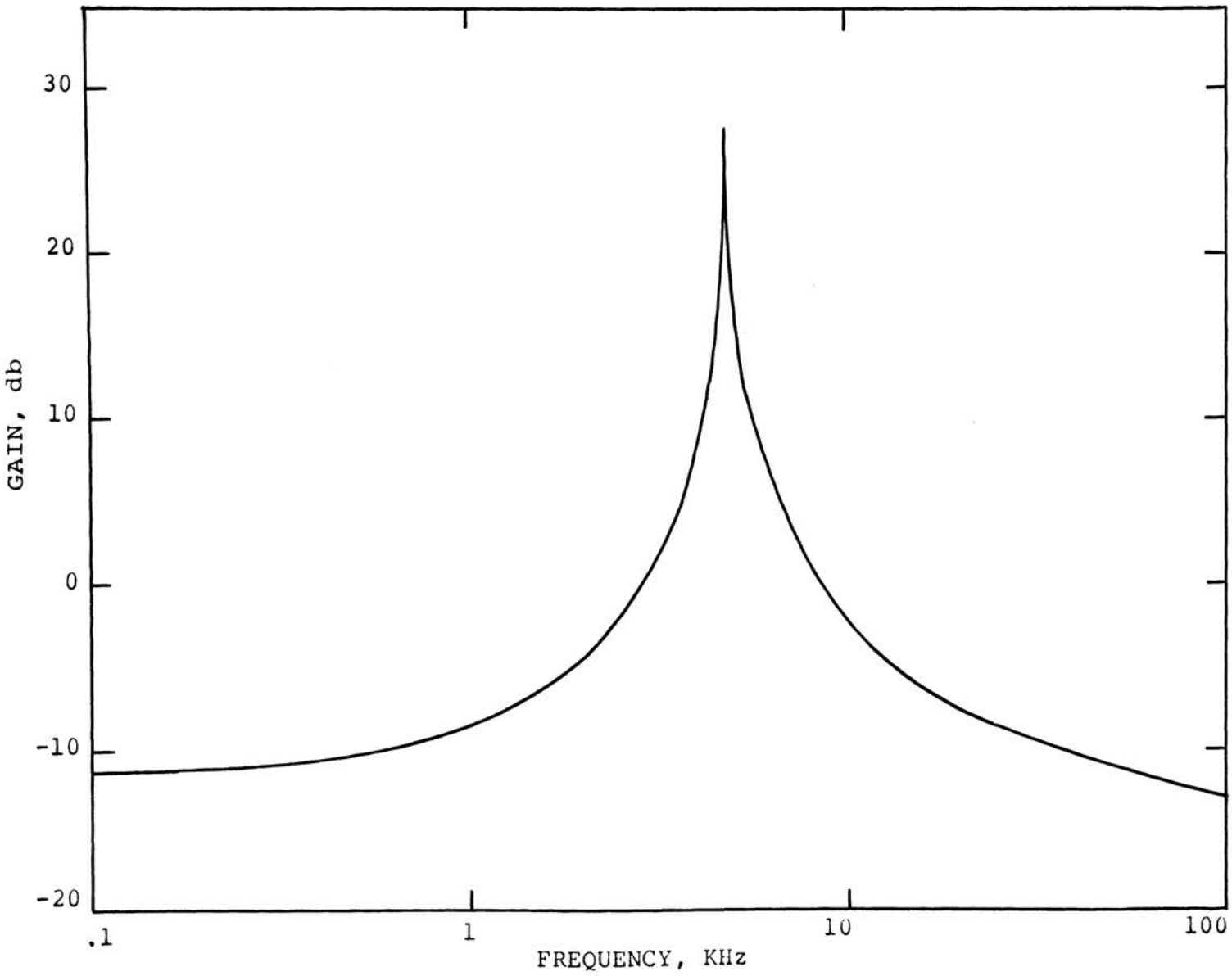

Figure 5. Gain vs. Frequency For The Model

approximation to the actual filter.

C. The Adjoint Circuit

Using the model of Figure 3 the adjoint network of Figure 6 is obtained by direct substitution of resistors, capacitors, and voltage controlled voltage sources for resistors, capacitors, and current controlled current sources respectively. Since the desired voltage responses occur at node seven of the original circuit, a current source, Φ, where

$$\Phi = (V_7^* - \hat{V}_7^*) \ ,$$

was applied to node seven of the adjoint network. Because there is no desired response at node eight, the voltage source there is set to zero.

Upon writing node voltage equations for the adjoint circuit a seven by seven network admittance matrix was obtained and is shown in Figure 7. Those elements which are underlined coincide exactly with the same elements of the original network admittance matrix. Therefore, many of the same steps used in reducing the original matrix are repeated in reducing the adjoint matrix. For this reason the original circuit solver subroutine, NETWK, through minor modifications, was also used to solve the adjoint circuit. This economy of programming steps allows for a considerable savings of storage space in the main program.

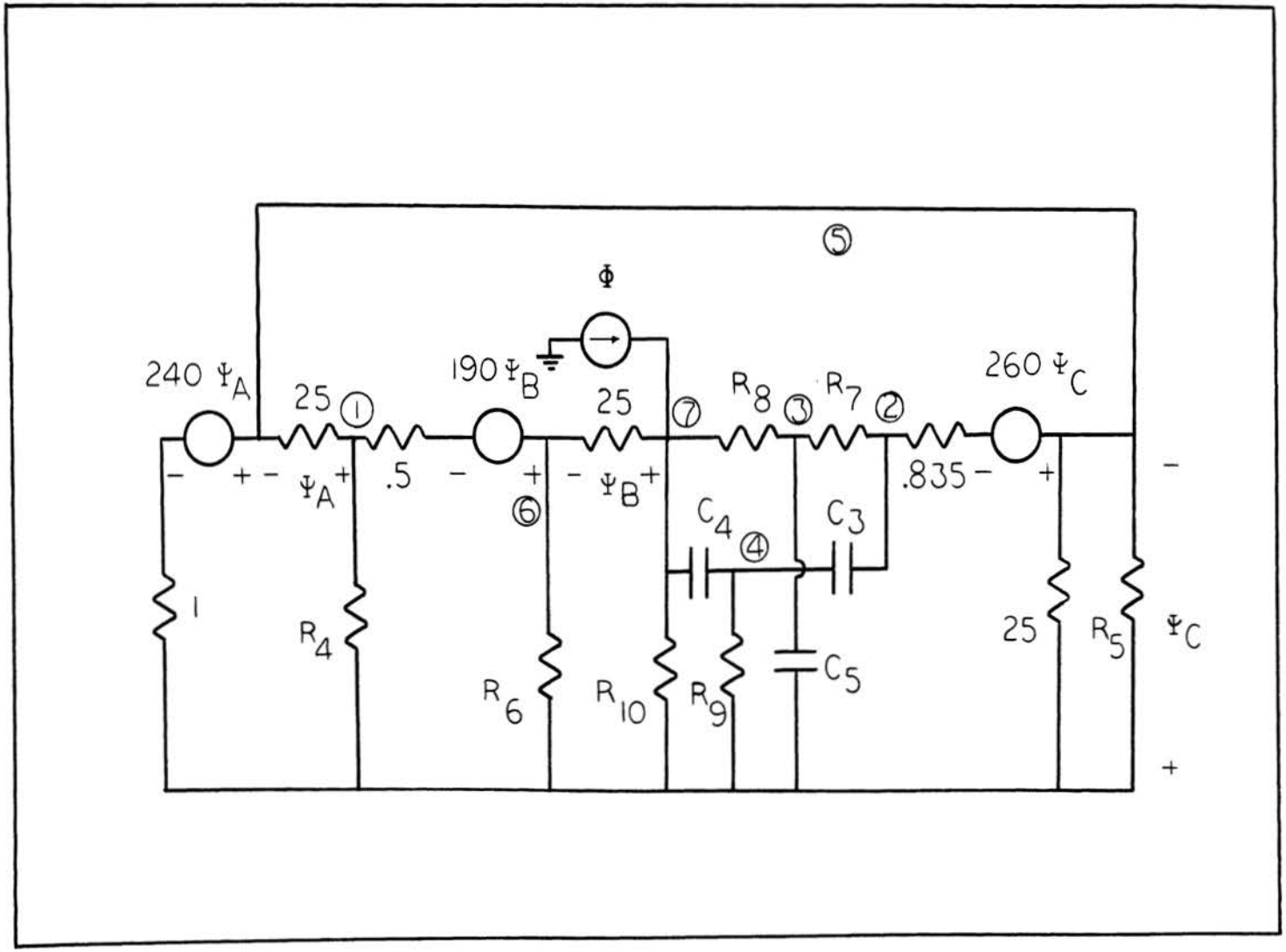

Figure 6. Adjoint Network Of Circuit Model

$$\begin{bmatrix} 11.64+G_4 & 0 & 0 & 0 & -9.64 & -392 & -392 \\ 0 & 1.2+G_7+j\omega C_3 & -G_7 & -j\omega C_3 & -313 & 0 & 0 \\ 0 & -G_7 & G_7+G_8+j\omega C_5 & 0 & 0 & 0 & -G_8 \\ 0 & -j\omega C_3 & 0 & G_9+j\omega(C_3+C_4) & 0 & 0 & -j\omega C_4 \\ -9.56 & -1.2 & 0 & 0 & 321.55 & 0 & 0 \\ -2 & 0 & 0 & 0 & 0 & 382.04+G_6 & -380.04 \\ 0 & 0 & -G_8 & -j\omega C & 0 & -.04 & .04+G_8+G_{10}+j\omega C_4 \end{bmatrix} \begin{bmatrix} \Psi_1 \\ \Psi_2 \\ \Psi_3 \\ \Psi_4 \\ \Psi_5 \\ \Psi_6 \\ \Psi_7 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

Figure 7. Nodal Admittance Matrix For Adjoint Circuit

V. THE PROGRAM

A. The Optimization Program

The optimization program (Appendix C) was designed so that any number and type of desired output voltages over any desired frequency range could be used as a criteria for optimization. The Fletcher-Powell optimization technique with a Fibonacci[4] search for α was used with a provision for using either the adjoint or perturbation method for calculating the network sensitivities. The program is general in every respect except for the network solving and gradient subroutines. These routines were written specifically for the given circuit in order to conserve computer time.

Once work was begun on the program it was soon apparent that if α were chosen such that $F(x + \alpha s)$ were a minimum, some of the circuit elements tried to assume negative values under certain conditions. While this might theoretically optimize the circuit, it does not produce a physical realization. This difficulty could be solved by techniques which impose a series of constraints upon the elements.[5] The problem was resolved in this case by allowing α to take on the maximum value possible before any of the elements became negative. Whenever this occurs the program output indicates which element or elements tend

toward a negative value.

Another problem occurred because α could not be found exactly. This can sometimes cause the error functional to increase from one iteration to the next. Whenever this occurs the program resets $\underline{H}$ to the identity matrix. This, in effect, is the method of steepest descent.[5]

Another feature incorporated into the program allows a one-dimensional search of the error surface to be made by varying any chosen element while holding all other elements constant. This procedure was found to be useful in getting a feel for the shape of the error functional surface.

The final optimization program consists of the main program, five subroutines and two functions. The subroutines are FLPOW, which carries out the Fletcher-Powell algorithm, GRAD, which solves for the error functional gradient vector, MIN, which finds a rough approximation to the α that minimizes $F(x + \alpha s)$, FIBMIN, which uses a Fibonacci search to minimize $F(x + \alpha s)$, and NETWK, which solves the original and the adjoint network equations for all node voltages. The functions are FX, which finds $F(x + \alpha s)$, and ERRF, which solves for the error functional. All operations are carried out in double precision. A flow chart of the final program is shown in Figure 8.

In order to test the effectiveness of the final optimization program, the optimization criteria of Figure 9

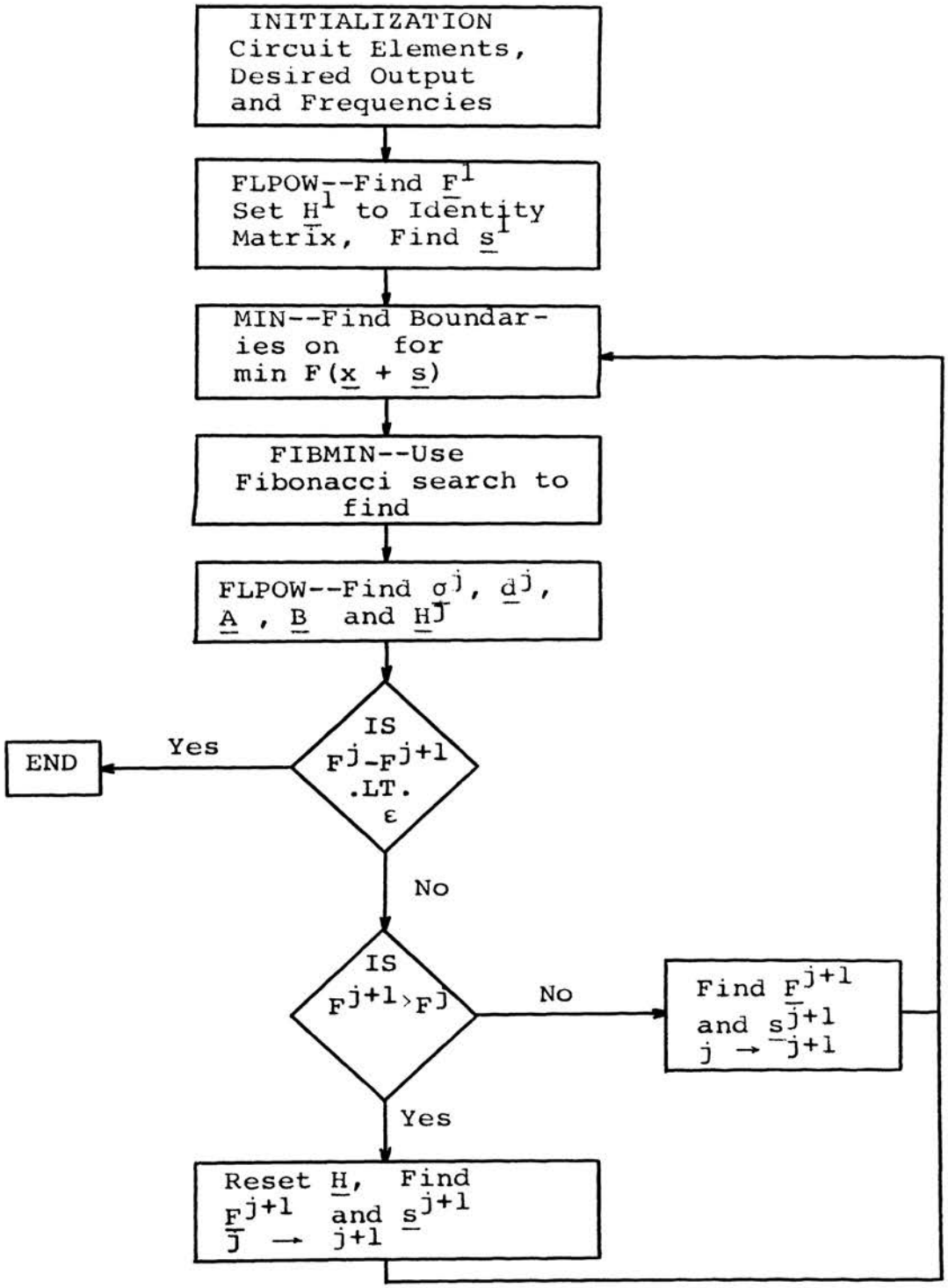

Figure 8. Flow Chart of Main Optimization Program

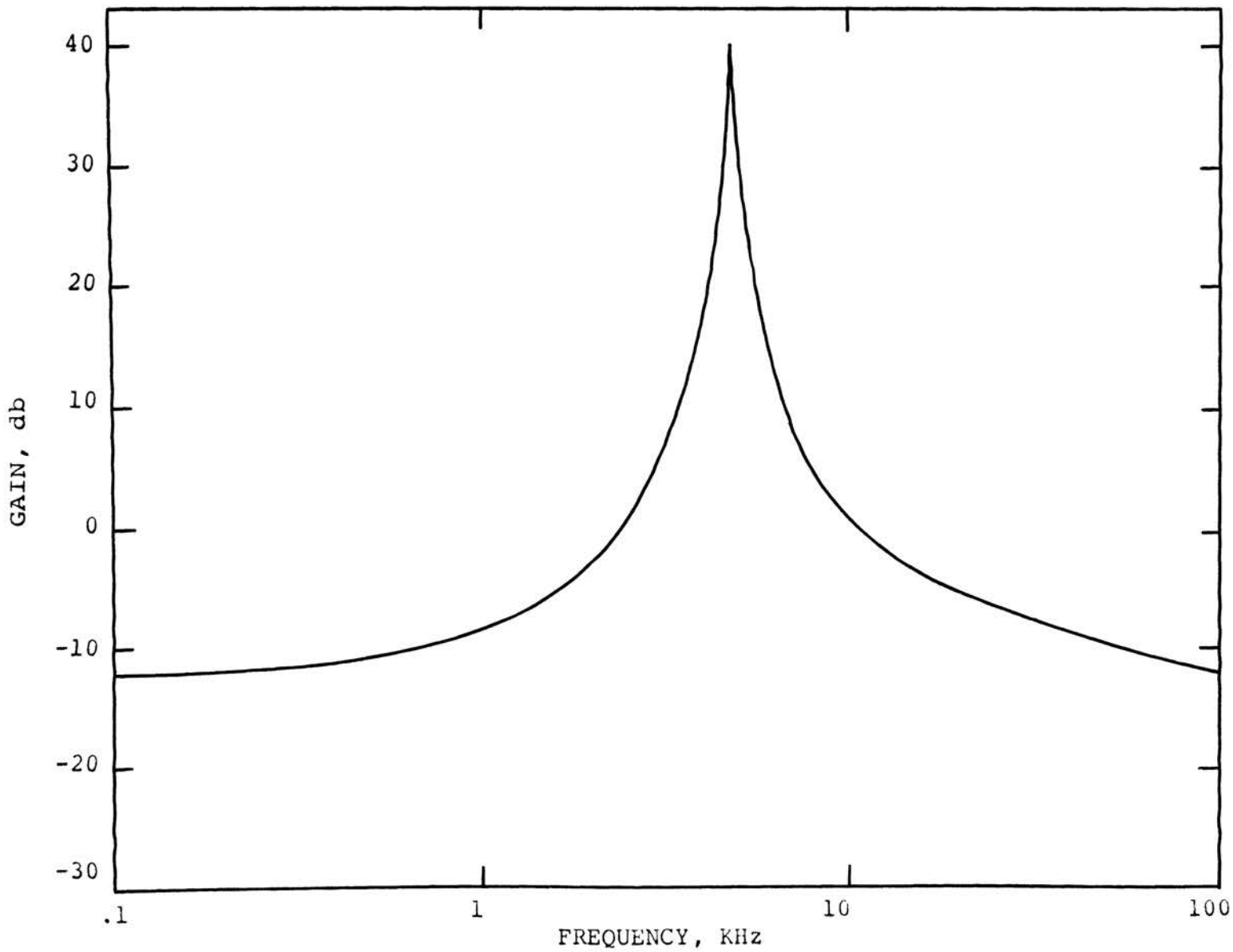

Figure 9. Final Results Of Perturbation And Adjoint Methods

was chosen. The actual gain of the filter at its center frequency was originally 24.2 decibels, or 17.2. As an illustrative example, a desired gain of 40 decibels, or 100, was specified. Figure 9 also shows the final results of both perturbation and adjoint methods. The computer time used for each method as well as some numerical results using the given criteria is listed in Table IV.

Figures 10, 11, 12, and 13 show the error functional as a function of some of the element values. It is interesting to note the relatively high sensitivity displayed with respect to the elements in the notch filter. If this large sensitivity were to cause programming problems, it could be reduced either by using a weighting function on the desired response or through the use of appropriate frequency scaling.[4]

B. The Circuit Analysis Program

In addition to the optimization program, a separate program to solve the network equations was written (Appendix B). This program solves for the seven node voltages, the voltage gain, the gain in decibels, and the phase angle in the frequency range from .1 kHz to 10 kHz for the network. The program is intended to facilitate plotting the transfer characteristics of the circuit for any given set of elements. Once the main program optimizes the circuit, the elements are read into the circuit solver program and all data

TABLE IV

TYPICAL RESULTS FROM THE FINAL OPTIMIZATION PROGRAM

	Method	
	Perturbation	Adjoint
Original Gain at Center Frequency	27.3	27.3
Original Value of Error Functional	5375.92	5375.92
Desired Gain at Center Frequency	100	100
Final Gain of Optimized Circuit at Center Frequency	99.9	99.8
Final Value of Error Functional	82.80	91.0
Computer Time Necessary for Optimization (Min.)	4:58	3:45
Average Number of Iterations of Fletcher-Powell Technique per Minute	1.05	1.40

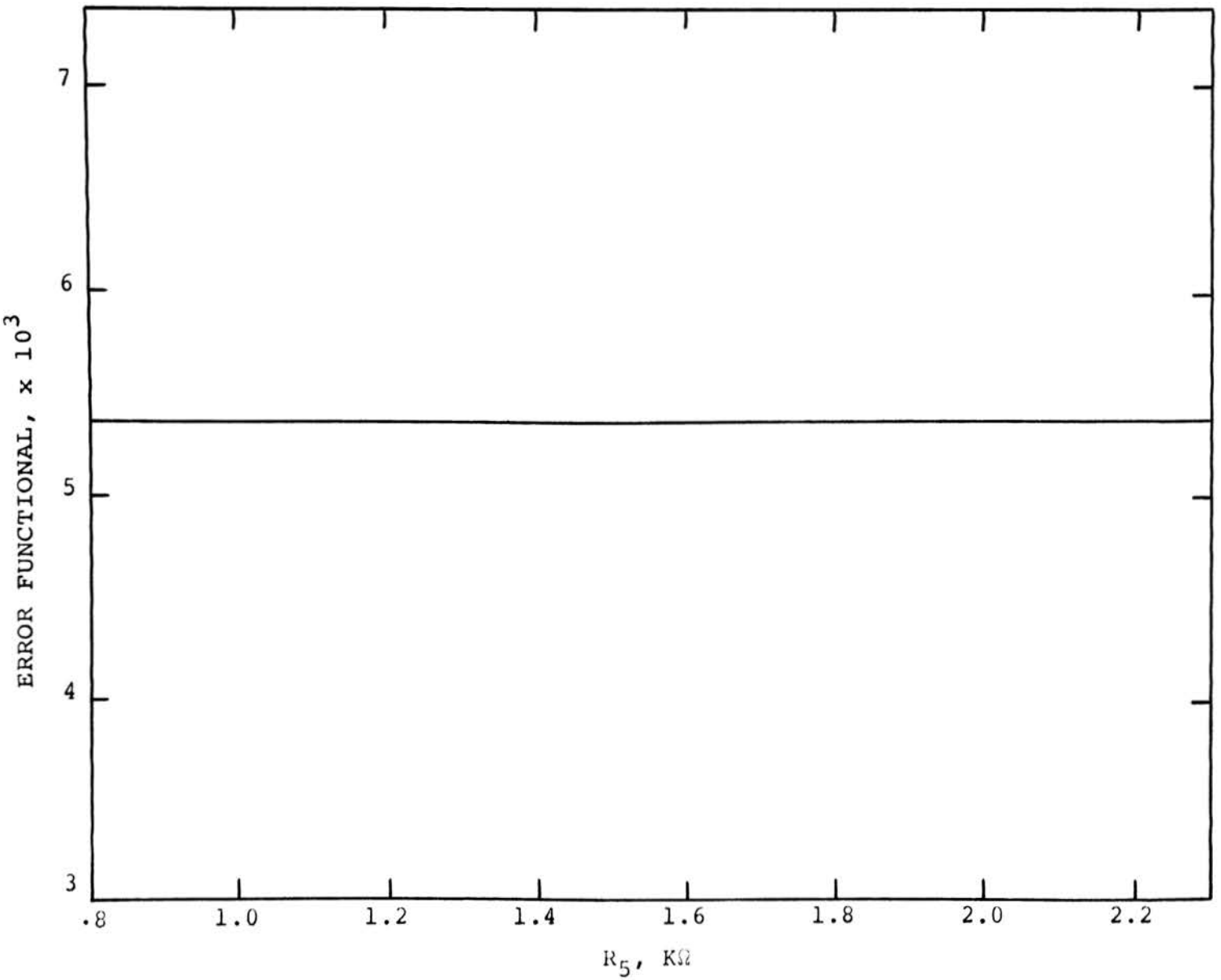

Figure 10. Error Functional vs. R_5

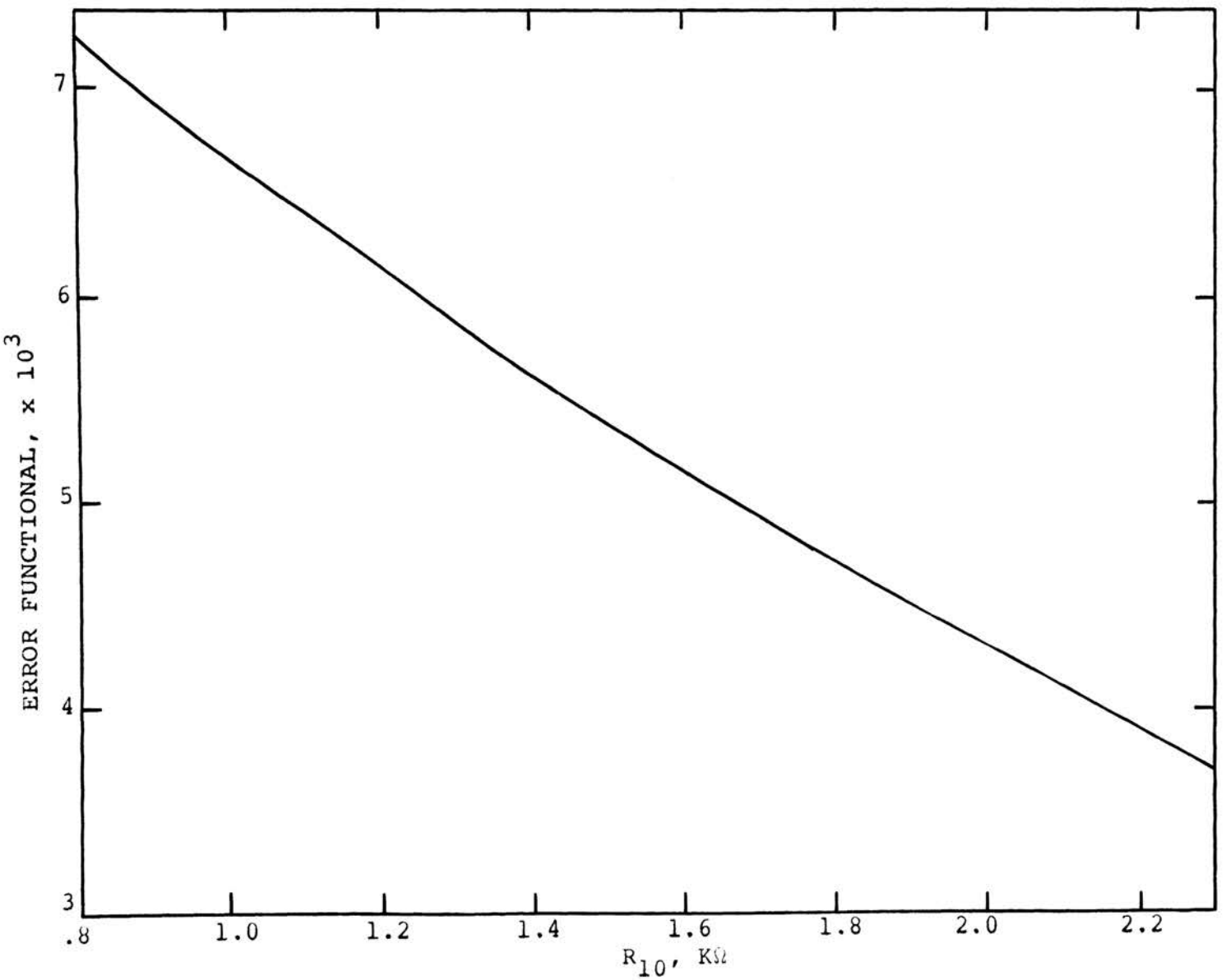

Figure 11. Error Functional vs. R_{10}

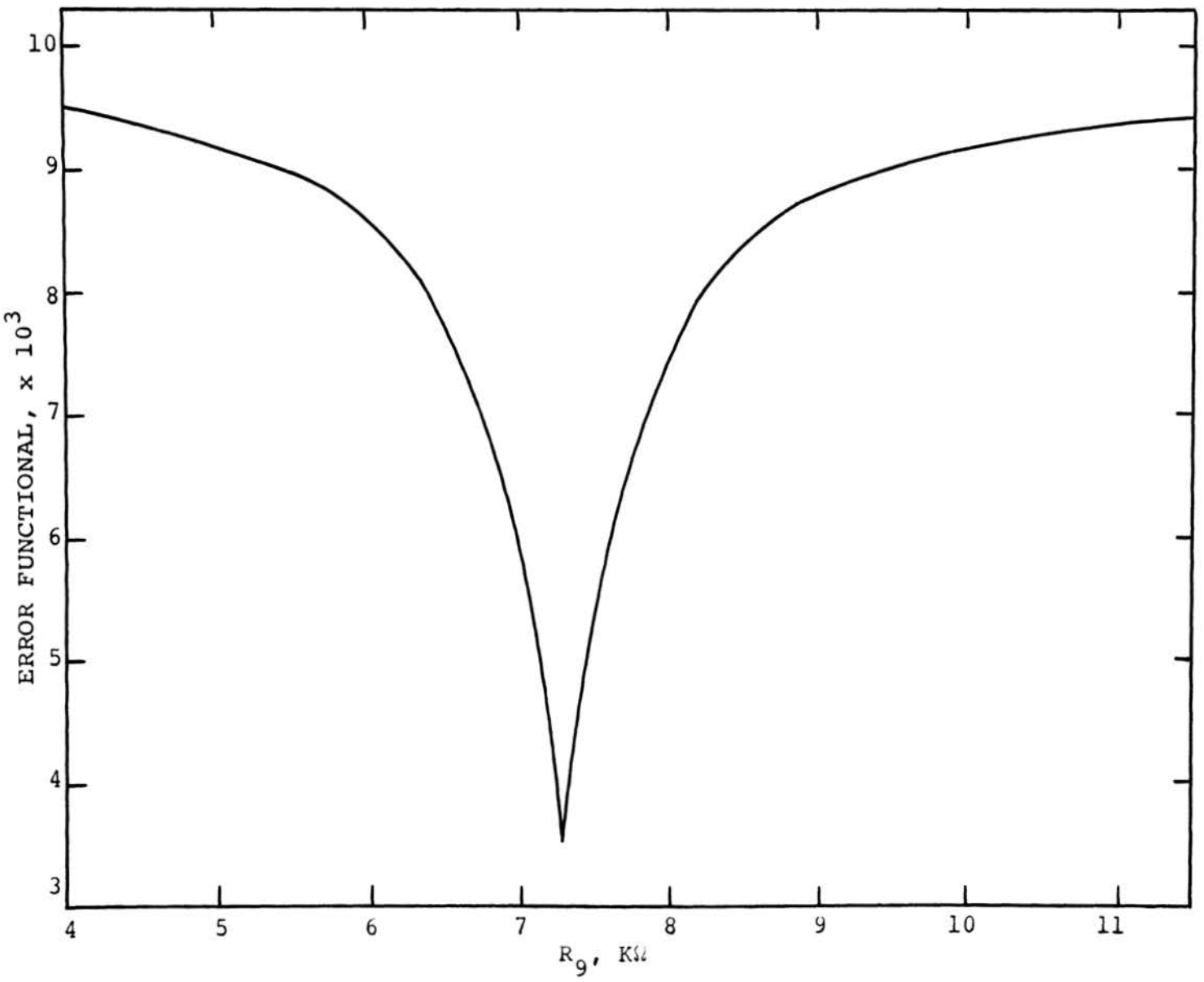

Figure 12. Error Functional vs. R_9

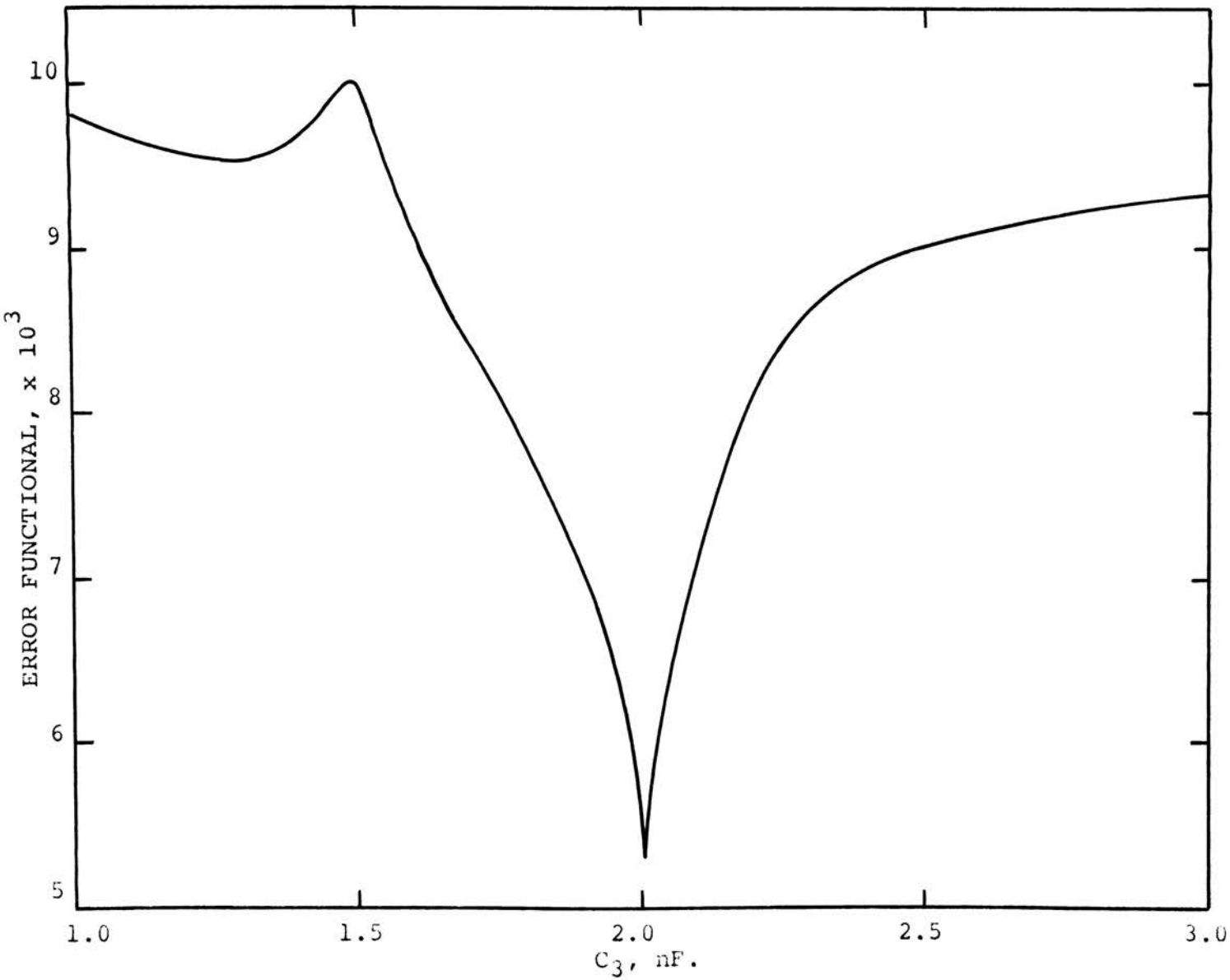

Figure 13. Error Functional vs. C_3

needed to plot the gain vs. log frequency curve is automatically generated. As in the optimization program, the circuit solver was written using double precision.

VI. CONCLUSIONS

A. Results

In writing the optimization program, one important conclusion was reached. As much time as possible should be spent in eliminating or simplifying as many steps in the program as possible. The program was constantly changed and improved as more features and added versatility were desired. The more general the program is made, the more computer time it will take; so if a generalized program is chosen for use, it will pay the programmer to spend the major portion of his time in streamlining the circuit solver portion of the program. This section is used far more often than any other during the optimization.

It should be apparent form the times listed in Table IV that the adjoint method has definite advantages over perturbation. The small amount of time lost in programming the adjoint method was compensated for by the reduction in computer time it made possible. The difference in computer time becomes even more dramatic as the number of circuit elements in increased.

B. Suggestions for Further Research

The adjoint method would be even more appealing if it could be implemented in a pre-packaged program, such

as ECAP, so that the adjoint network was automatically formed from the original. This should be relatively easy to accomplish since the adjoint network's topology is unchanged. A problem might be encountered when using controlled sources. The adjoint element for a current controlled current source is a voltage controlled voltage source, and this element would have to be converted to an element recognized by ECAP. Further work could most certainly be done in this direction.

BIBLIOGRAPHY

1. S. W. Director and R. A. Rohrer, "Automated Network Design--The Frequency Domain Case", IEEE Trans. Circuit Theory, vol. CT-16, pp. 330-337, September, 1967.

2. R. A. Rohrer, "Fully Automated Network Design by Digital Computer: Preliminary Considerations", Proc. IEEE, vol. 55, pp. 1929-1939, November, 1967.

3. S. W. Director and R. A. Rohrer, "The Generalized Adjoint Network and Network Sensitivities", IEEE Trans. Circuit Theory, vol. CT-16, pp. 318, 323, September, 1967.

$. G. C. Temes and D. A. Calahan, "Computer-Aided Optimization--The State of the Art", Proc. IEEE, vol. 55, pp. 1832-1863, November, 1967.

5. D. A. Calahan, Computer-Aided Network Design. New York: McGraw-Hill, 1968.

6. N. Sato and W. F. Tinney, "Techniques for Exploiting the Sparsity of the Network Admittance Matrix", IEEE Trans. on Power and Apparatus, vol. 82, pp. 944-950, December, 1963.

7. C. A. Desoer and E. S. Kuh, Basic Circuit Theory, vol. 1, New York: McGraw-Hill, 1967, pp. 292-300.

8. R. C. Daleman, "Narrow-Band High-Q Filter Uses Passive Twin-T", Electronic Design, vol. 13, p. 110, June, 1968.

9. R. Fletcher and M. J. D. Powell, "A Rapdily Convergent Descent Method for Minimization", The Computer Journal, vol. 6, pp. 163-168, 1963.

10. H. H. Rosenbrock, "An Automatic Method for Finding the Greatest or Least Value of a Function", The Computer Journal, vol. 3, pp. 175-184, 1960.

11. "Electronic Circuit Analysis Program (ECAP)" IBM Corporation, DAta Processing Div., White Plains, N. Y., Application Program 360-E.

12. W. F. Tinney and J. W. Walker, "Direct Solutions of Sparse Network Equations by Optimally Ordered Triangular Factorization", Proc. IEEE, vol. 55, pp. 1801-1809, November, 1967.

13. D. F. Dawson, F. F. Kuo, and W. G. Magnuson, Jr., "Computer-Aided Design of Electronic Circuits--A User's Viewpoint", Proc. IEEE, Vol. 55, pp. 1946-1954, November, 1967.

14. L. Stanton, "Theory and Application of Parallel-T Resistance-Capacitance Frequency-Selective Networks", Proc. IRE, Vol. 34, pp. 447-456, 1946.

VITA

John Walter Olson was born on December 27, 1949, in Springfield, Missouri. He received his primary and secondary education in Ava, Missouri, graduating valedictorian of his high school class in May, 1967. He entered the University of Missouri - Rolla, in Rolla, Missouri in September, 1967, and received a Bachelor of Science Degree in Electrical Engineering in May, 1971. During this time he was the recipient of a four year ROTC scholarship and a Curators' Scholarship.

He has been enrolled in the Graduate School of the University of Missouri - Rolla since June, 1971.

The author is a member of Tau Beta Pi, Phi Kappa Phi and Eta Kappa Nu and has been a commissioned officer in the United States Army Military Intelligence Branch since May, 1971.

APPENDIX A

Standard Matrix Inversion Subroutine Used to Determine The Effectiveness of the Network Analysis Program

```
      SUBROUTINE MATINV (A,N,B,M,DETERM)
      DIMENSION A(10,10),B(10,10),INDEX(10,2),PIVOT(10),
           IPIVOT(10)
      COMPLEX A,B,AMAX,SWAP,PIVOT,DETERM,T
      COMPLEX AIC
   10 DETERM=(1.0,0.0)
   15 DO 20 J=1,N
   20 IPIVOT(J)=0
   30 DO 550 I=1,N
   40 AMAX=(0.0,0.0)
   45 DO 105 J=1,N
   50 IF(IPIVOT(J)-1)  60,105,60
   60 DO 100 K=1,N
   70 IF(IPIVOT(K)-1)  80,100,740
   80 IF(CABS(AMAX)-CABS(A(J,K)))  85,100,100
   85 IROW=J
   90 ICOLUM=K
   95 AMAX=A(J,K)
  100 CONTINUE
  105 CONTINUE
  110 IPIVOT(ICOLUM)=IPIVOT(ICOLUM)+1
  130 IF(IROW-ICOLUM)  140,260,140
  140 DETERM=-DETERM
  150 DO 200 L=1,N
  160 SWAP=A(IROW,L)
  170 A(IROW,L)=A(ICOLUM,L)
  200 A(ICOLUM,L)=SWAP
  205 IF(M)  260,260,210
  210 DO 250L=1,M
  220 SWAP=B(IROW,L)
  230 B(IROW,L)=B(ICOLUM,L)
  250 B(ICOLUM,L)=SWAP
  260 INDEX(I,1)=IROW
  270 INDEX(I,2)=ICOLUM
  310 PIVOT(I)=A(ICOLUM,ICOLUM)
  320 DETERM=DETERM*PIVOT(I)
  330 A(ICOLUM,ICOLUM)=(1.0,0.0)
  340 DO 350 L=1,N
  350 A(ICOLUM,L)=A(ICOLUM,L)/PIVOT(I)
  355 IF(M)  380,380,360
  360 DO 370 L=1,M
  370 B(ICOLUM,L)=B(ICOLUM,L)/PIVOT(I)
  380 DO 550 L1=1,N
  390 IF(L1-ICOLUM)  400,550,400
  400 T=A(L1,ICOLUM)
  420 A(L1,ICOLUM)=(0.0,0.0)
  430 DO450 L=1,N
```

```
450   A(L1,L)=A(L1,L)=A(ICOLUM,L)*T
455   IF(M) 550,550,460
460   DO 500 L=1,M
500   B(L1,L)=B(L1,L)-B(ICOLUM,L)*T
550   CONTINUE
600   DO 710 I=1,N
610   L=N+1-I
620   IF(INDEX(L,1)-INDEX(L,2)) 630,710,630
630   JROW=INDEX(L,1)
640   JCOLUM=INDEX(L,2)
650   DO 705 K=1,N
660   SWAP=A(K,JROW)
670   A(K,JROW)=A(K,JCOLUM)
700   A(K,JCOLUM)=SWAP
705   CONTINUE
710   CONTINUE
740   RETURN
      END
```

APPENDIX B

The Network Analysis Program

```
C
C  NETWORK SOLVER
C
      IMPLICIT REAL*8(A-H,0-Z)
      DIMENSION V(10)
      COMMON X(12),DESIRE(10),WDES(10),XSWAP(12),NEL
      COMPLEX*16 V
      COMPLEX*8 VP7P
      REAL*4 ANGLE,AT AN2,AIMAG, REAL
      READ(1,1) NEL,NI
    1 FORMAT (I2)
      DO 1000 L=1,NI
      WRITE(3,5)
    5 FORMAT('1')
      READ(1,2)  (X(I),I-1,NEL)
    2 FORMAT(5F10.3)
      WRITE(3,6)  (X(I),I=1,NEL)
    6 FORMAT(T10,'X = ',6D15.6,/)
      KEY=0
      DO 100 I=1,4
      DO 100 J=1,9
      F-(10.D)**I)*J/100000.DO
      W=2.*3.14159*F
      F=F*1.D+6
      IF(I.NE.4)GO TO 7
      IF(J.EQ.1)GO TO 7
      F=.005308DO
      W=2.*3.14159*F
      F=F*1.D+6
      KEY=1
    7 CALL NETWK1(V,W,3)
      WRITE(3,3) (V(M),M=1,7)
    3 FORMAT(T60,7D9.2,/)
      VM=CDABS(V(7))
      DB=20.*DLOG10(VM)
      VP7P=V(7)
      ANGLE=ATAN2(AIMAG(VP7P),REAL(VP7P))*180./3.14159
      WRITE(3,4) F,VM,DB,ANGLE
    4 FORMAT(T10,'F = ',D10.2,T30,'G = ',D10.3,T50,'GDB =
     '  ',D10.3,1T70,'ANGLE = ',E10.3,/)
      IF(KEY.EQ.1) GO TO 1000
  100 CONTINUE
 1000 CONTINUE
      STOP
      END

      SUBROUTINE NETWK1(V,W,KEY)
      IMPLICIT REAL*B(A-H,0-Z)
```

```
      DIMENSION Y(8,8),V(10),X(12)
      COMPLEX*16 PIVOT,Y,V,DCMPLX
      COMMON X,DESIRE(10),WDES(10),XSWAP(12),NEL
      G2=1./X(1)
      G3=1./X(2)
      G4=1./X(3)
      G5=1./X(4)
      G6=1./X(5)
      G7=1./X(6)
      G8=1./X(7)
      G9=1./X(8)
      G10=1./X(9)
      C3=X(10)
      C4=X(11)
      C5=X(12)
      DO 7L=1,8
      V(L)=(0.DO,0.DO)
      DO 7 M=1,8
      Y(L,M)=(0.DO,0.DO)
 7    CONTINUE
      Y(1,1)=DCMPLX(2.04+G4,0.DO)
      Y(1,5)=(-9.46DO,0.DO)
      Y(1,6)=(-2.DO,0.DO)
      Y(1,8)=(-2.30384DO,0.DO)
      Y(2,2)=DCMPLX(1.2+G7,W*C3)
      Y(2,3)=DCMPLX(-G7,0.DO)
      Y(2,4)=DCMPLX(0.DO,=W*C3)
      Y(2,5)=(-1.2DO,0.DO)
      Y(3,2)=Y(2,3)
      Y(3,3)=DCMPLX(G7+G8,W*C5)
      Y(3,7)=DCMPLX(-G8,0.DO)
      Y(4,2)=Y(2,4)
      Y(4,4)=DCMPLX(G9,W*(C4+C3))
      Y(4,7)=DCMPLX(0.DO,=W*C4)
      Y(5,1)=(-.04DO,0.DO)
      Y(5,2)=(-312.DO,0.DO)
      Y(5.5)=DCMPLX(321.55DO+G5,0.DO)
      Y(5,8)=(2.3111DO,0.DO)
      Y(6,1)=(-382.DO,0.DO)
      Y(6,6)=DCMPLX(382.04+G6,0.DO)
      Y(6,7)=(-.04DO,0.DO)
      Y(7.1)=(380.D0,0.D0)
      Y(7,3)=Y(3,7)
      Y(7,4)=Y(4,7)
      Y(7,6)=(-380.04D0,0.D0)
      Y(7,7)=DCMPLX(.04+G8+G10,W*C4)
      PIVOT=Y(1,1)
      D0 10 I=5,8
10    Y(1,I)=Y(1,I)/PIVOT
      Y(5,8)=Y(5,8)=Y(5,1)*Y(1,8)
      Y(5,6)=-Y(1,6)*Y(5,1)
      Y(5,5)=Y(5,5)=Y(1,5)*Y(5,1)
```

```
      Y(6,8)=-Y(6,1)*Y(1,*)
      Y(6,6)=Y(6,6)=Y(6,1)*Y(1,6)
      Y(6,5)=-Y(6,1)*Y(1,5)
      Y(7,8)=-Y(7,1)*Y(1,8)
      Y(7,6)=Y(7,6)-Y(7,1)*Y(1,6)
      Y(7,5)=-Y(7,1)*Y(1,5)
      PIVOT=Y(2,2)
      D0 20 I=3,5
20    Y(2,I)=Y(2,I)/PIVOT
      Y(3,5)=-Y(3,2)*Y(2,5)
      Y(3,4)=-Y(3,3)*Y(2,4)
      Y(3,3)=Y(3,3)=Y(3,2)*Y(2,3)
      Y(4,5)=-Y(4,2)*Y(2,5)
      Y(4,4)=Y(4,4)=Y(4,2)*Y(2,4)
      Y(4,3)=-Y(4,2)*Y(2,3)
      Y(5,5)=Y(5,5)-Y(5,2)*Y(2,5)
      Y(5,4)=-Y(5,2)*Y(2,4)
      Y(5,3)=-Y(5,2)*Y(2,3)
      PIVOT=Y(3,3)
      D0 30 I=4,7
30    Y(3,I)=Y(3,I)/PIVOT
      Y(4,7)=Y(4,7)-Y(4,3)*Y(3,7)
      Y(4,5)=Y(4,5)-Y(4,3)*Y(3,5)
      Y(4,4)=Y(4,4)-Y(4,3)*Y(3,4)
      Y(5,7)=-Y(5,3)*Y(3,7)
      Y(5,5)=Y(5,5)-Y(5,3)*Y(3,5)
      Y(5,4)=Y(5,4)-Y(5,3)*Y(3,4)
      Y(7,7)=Y(7,7)-Y(7,3)*Y(3,7)
      Y(7,5)=Y(7,5)-Y(7,3)*Y(3,5)
      Y(7,4)=Y(7,4)-Y(7,3)*Y(3,4)
      Y(4,5)=Y(4,5)/Y(4,4)
      Y(4,7)=Y(4,7)/Y(4,4)
      Y(5,7)-Y(5,7)-Y(5,4)*Y(4,7)
      Y(5,5)=Y(5,5)-Y(5,4)*Y(4,5)
      Y(7,7)=Y(7,7)-Y(7,4)*Y(4,7)
      Y(7,5)=Y(7,5)-Y(7,4)*Y(4,5)
      D0 40 I=6,8
40    Y(5,I)=Y(5,I)/Y(5,5)
      Y(6,8)=Y(6,8)-Y(6,5)*Y(5,8)
      Y(6,7)-Y(6,7)-Y(6,5)*Y(5,7)
      Y(6,6)-Y(6,6)-Y(6,5)*Y(5,6)
      Y(7,8)=Y(7,8)-Y(7,5)*Y(5,8)
      Y(7,7)=Y(7,7)-Y(7,5)*Y(5,7)
      Y(7,6)=Y(7,6)-Y(7,5)*Y(5,6)
      Y(6,7)=Y(6,7)/Y(6,6)
      Y(6,8)-Y(6,8)/Y(6,6)
      Y(7,8)=Y(7,8)-Y(7,6)*Y(6,8)
      Y(7,7)=Y(7,7)-Y(7,6)*Y(6,7)
      Y(7,8)=Y(7,8)/Y(7,7)
      V(7)=Y(7,8)
      IF(KEY.EQ.1)GO TO 50
      V(6)=Y(6,8)-Y(6,7)*V(7)
```

```
      V(5)=Y(5,8)-Y(5,7)*V(7)-Y(5,6)*V(6)
      V(4)=-Y(4,5)*V(5)-Y(4,7)*V(7)
      V(3)=-Y(3,4)*V(4)-Y(3,5)*V(5)-Y(3,7)*V(7)
      V(2)=-Y(],2)*V(3)-Y(2,4)*V(4)-Y(2,5)*V(5)
      V(1)=Y(1,8)-Y(1,5)*V(5)-Y(1,6)*V(6)
50    RETURN
      END
```

APPENDIX C

The Final Optimization Program

```
C
C  KEY = 2 PERTURBATION METHOD USED
C  KEY = 3 ADJOINT METHOD USED
C  KEY = 4 A SCAN OF THE ERROR SURFACE IS MADE
C          VARRYING ONLY ONE ELEMENT
C
      IMPLICIT REAL*8(A-H,0-7)
      DIMENSION X(12),DESIRE(30),WDES(30),XSWAP(12)
      COMMON X,DESIRE,WDES,XSWAP,NEL,NW
C*****READ NUMBER OF ELEMENTS AND NUMBER OF TEST POINTS
      READ(1,1)NEL,NW
    1 FORMAT(12)
C*****READ DESIRED OUTPUT AND FREQUENCIES
      READ (1,2) (DESIRE(I),I=1,NW),(WDES(J),J=1,NW)
    2 FORMAT(5F10.3)
C*****READ ELEMENT VALUES
      READ(1,3)  (X(I),I=1,NEL)
    3 FORMAT(5,F10.3)
      WRITE(3,4)
      WRITE(3,6)  (DESIRE(I),I=1,NW)
      WRITE(3,7)  (WDES(J),J=1,NW)
      WRITE(3,8)
    4 FORMAT('1',T43,45(1H*),/,T43,'* OPTIMIZATION OF AN
           ACTIVE BANDPASS 1 FILTER*',/,T43,45(1H*),//T10,
           110(1H*))
    6 FORMAT(//,T15,'DESIRED OUTPUT VOLTAGES',/,T20,10F10.3)
    7 FORMAT(//,T15,'FREQUENCIES IN MHZ',/,T20,10F10.4)
    8 FORMAT(//,T10,110(1H*),//)
C*****CONVERT FREQUENCIES TO RADIAN FREQUENCY
      D0 9 L=1,NW
      WDES(L)=2.*3.14159*WDES(L)
    9 CONTINUE
C*****READ KEY
      READ(1,1) KEY
C*****ADJOINT METHOD
      IF(KEY.EQ.3)WRITE(3,10)
   10 FORMAT(655,'ADJOINT METHOD USED',//)
C*****PERTURBATION METHOD
      IF KEY.EQ.2)WRITE(3,11)
   11 FORMAT(T53,'PERTURBATION METHOD USED'//)
C*****ERROR SCAN
      IF(KEY.NE.4)GO TO 13
C*****READ NUMBER OF SCANS AND ELEMENTS TO BE VARRIED
      READ (1,1) NKEY
      D0 16 II=1,NKEY
      READ(1,1) LKEY
      WRITE(3,14) LKEY
```

```
   14 FORMAT(1H1,T20,12HELEMENT NO. , 12,11H IS VARIED,//)
      DEL=.02*X(LKEY)
      SWAP=H(LKEY)
      X(LKEY)=X(LKEY)-26.*DEL
      DO 5 L=1,50
      X(LKEY)=X(LKEY)+DEL
      F=ERRF(1)
      WRITE(3,12) X(LKEY),F
    5 CONTINUE
      X(LKEY)=SWAP
   16 CONTINUE
      STOP
   13 CONTINUE
   12 FORMAT(T20,'X(5) - ',D15.6,' F = ',D15.6,/)
      WRITE(3,15)
   15 FORMAT(//,T56,'THE ELEMENTS ARE:',//,T20,'X =
           ',T34,'R2',T48,'R3',1T65,'R4',T79,'$5',T95,
           'R6',T109,'R7',//,T20,'X' =',T34,'R8',T48,
           2'R9',T65,'R10',T79,'C3',T95,'C4',T109,'C5',///)
      CALL FLPOW(KEY)
      STOP
      END
C
      SUBROUTINE FLPOW(KEYADJ)
C*****THIS SUBROUTINE CARRIES OUT THE FLETCHER-POWELL
C*****OPTIMIZATION TECHNIQUE
      IMPLICIT REAL*8(A-H,0-Z)
      DIMENSION X(12,H(12,12),S(12),GR(12),WDES(30),
           DESIRE(30),1SIG(12),GR1(12),D(12),HD(12),DH(12)
      COMMUN X,DESIRE,WDES,DH,NEL,NW
      WRITE(3,4) (X(I),I=1,NEL
C*****THE H MATRIX IS SET INITIALLY TO THE IDENTITY MATRIX
      D0 3 I=1,NEL
      D0 2J=1,NEL
    2 H(I,J)=0.D0
    3 H(I,I)=1.D0
C*****CALCULATE AND WRITE THE ERROR FUNCTIONAL
      F=ERRF(1)
      ERX=F
      WRITE(3,5) F
C*****CALCULATE THE NETWORK SENSITIVITIES
      CALL GRAD(GR,F,KEYADJ)
C*****FORM THE S VECTOR
    1 D0 10 I=1,NEL
      S(I)=0.D0
      D0 10 J=1,NEL
      S(I)=S(I)-H(I,J)*GR(J)
   10 CONTINUE
C*****MINIMIZE F(X + ALPHA*S)
      CALL MIN(ALF,S,F)
      WRITE(3,6)ALF
    6 FORMAT(/,T53,ALPHA = ',D15.6,/,T53,23(1H-),//)
C*****CALCULATE SIGMA
```

```
      DO 20 I=1,NEL
   20 X(I)=X(I)+SIG(I)
C*****WRITE NEW ELEMENT VALUES AND NEW VALUES OF ERROR
           FUNCTIONAL
      WRITE(3,4)  (X(I),I=1,NEL)
    4 FORMAT(T20,'X = ',6F15.5,/)
      ERX1=F
      WRITE(3,5) ERX1
C*****CALCULATE NEW SENSITIVITIES
      CALL GRAD(GR1,F,KEYADJ)
C*****FORM THE D MATRIX
      DO 30 I=1,NEL
   30 D(I)=GR1(I)-GR(I)
      DENA=0.D0
      DO 40 I=1,NEL
      GR(I)=GR1(I)
   40 DENA-DENA+SIG(I)*D(I)
      DO 60 I=1,NEL
      HD(I)=0.D0
      DO 60 J=1,NEL
   60 HD(I)=HD(I)+H(I,J)*D(J)
      DENB=0.D0
      DO 70 I=1,NEL
   70 DENB=DENB+HD(I)*D(I)
      DO 90 I=1,NEL
      DH(I)=0.D0
      DO 90 J=1,NEL
   90 DH(I)=DH(I)+D(J)*H(J,I)
C*****FIND THE NEW H MATRIX
      DO 100 I=1,NEL
      DO 100 J=1,NEL
  100 H(I,J)=H(I,J)+SIG(I)*SIG(J)/DENA=HD(I)*DH(J)/DENB
    5 FORMAT(/,T48,'ERROR FUNCTIONAL=',D15.6,/,T48,34
           (1H-),//)
C*****IF THE ERROR FUNCTIONAL IS NOT DECREASED RESET
           THE H MATRIX
      IF(ERX1.LT.ERX)GO TO 7
      DO 120 I=1,NEL
      DO 121 J=1,NEL
  121 H(I,J)=0.D0
  120 H(I,I)=1.D0
C*****TEST FOR CONVERGENCE
    7 IF(DABS(ERX1-ERX).LE.1.D-5)GO TO 2000
      ERX=ERX1
      GO TO 1
 2000 WRITE(3,4) (X(I),I=1,NEL)
      WRITE(3,5)F
      WRITE(3,101)
  101 FORMAT(///,T10,27(4HEND ),/,'1')
      RETURN
      END
```

```
C
      SUBROUTINE GRAD(GR,F,KEYADJ)
C*****THIS SUBROUTINE FINDS THE GRADIENT VECTOR USING
           PERTURBATION
C*****IF KEYADJ = 3 THE GRADIENT VECTOR IS FOUND USING
           THE ADJOINT METHOD
      IMPLICIT REAL*8(A-H,0-Z)
      DIMENSION GR(12),X(12),DESIRE(30),WDES(30),PSI
          (8),V(10),XSWAP(12)1,GS(12)
      COMMON X,DESIRE,WDES,XSWAP,NEL,NW
      COMPLEX*16 V,PSI,DCMPLX, GS,DCONJG
      COMPLEX*8GX
      REAL*4 REAL
C*****TEST TO SEE WHICH METHOD IS USED
      IF(KEYADJ.NE.3)GO TO 2
C
C*****ADJOINT METHOD
C
      D0 5 I=1,NEL
    5 GR(I)=0.D0
      D0 4 I=1,NW
      W=WDES(I)
      PSI(8)=DCMPLX(DESIRE(I),0.D0)
      CALL NETWK(V,W,3,PSI)
      GS(3)=(V(1)/X(3))*(PSI(1)/X(3))
      GS(4)=V(5)/X(4))*(PSI(5)/X(4))
      GS(5)=-(V(6)/X(5))*(PSI(6)/X(5))
      GS(6)=((V(2)-V(3))/X(6))*((PSI(2)-PSI(3))/X(6))
      GS(7)=((V(3)=V(7))/X(7))*((PSI(3)-PSI(7))/X(7))
      GS(8)=(V(4)/X(8))*(PSI(4)/X(8))
      GS(9)=-(V(7)/X(9))*(PSI(7)/X(9))
      GS(10)=DCMPLX(0.D0,W)*(V(3)-V(2))*(PSI(3)-PSI(2))
      GS(11)=DCMPLX(0.D0,W)*(V(7)-V(3))*(PSI(7)-PSI(3))
      GS(12)=DCMP(0.D0,W)*V(3)*PSI(3)
      D0 7 J=3,9
      GS(1)=(GS(J)+DCONJG(GS(J)))*.5D0
      GX=GS(1)
      GR(J)=GR(J)-DSIGN(CDABS(GS(1)),DBLE(REAL(GX)))
    7 CONTINUE
      D0 4 J=10,12
      GS(1)=(GS(J)+DCONJG(GS(J)))*.5D0
      GX=GS(1)
      GR(J)=GR(J)+DSIGN(CDABS(GS(1)),DBLE(REAL(GX)))
    4 CONTINUE
      GO TO 3
C
C*****PERTURBATION METHOD
C
    2 ERR=F
      D0 10 I=1,NEL
      SWAP=X(I)
      DELX=.001*X(I)
```

```
      X(I)=X(I)+DELX
      ERDEL=ERRF(1)
      X(I)=SWAP
      GR(I)=(ERDEL-ERR)/DELX
   10 CONTINUE
C*****WRITE THE ELEMENT SENSITIVITIES
    3 WRITE(3,1) (GR(I), I=1,NEL)
    1 FORMAT(T20,'G = ',6D15.5)
      WRITE(3,6)
    6 FORMAT(/,T10,110(1H-))
      RETURN
      END
C
      SUBROUTINE MIN(ALE,S,F)
C*****THIS SUBROUTINE MINIMIZES F(X+ALF*S)
      IMPLICIT REAL*8(A-H,0-Z)
      DIMENSION S(12),X(12),DESIRE(30),WDES(30),X1(12)
      COMMON X,DESIRE,WDES,S1,NEL,NW
      R=2.D0
      FB=1.D20
      FC=F
      ALFM2=0.D0
      ALFM1=1.D-20
      ALFK=1.D-8
      D0 10 K=1,100
      FA=FB
      FB=FC
      FC=FX(ALEK,S,KEY)
C*****IF ONE OF THE ELEMENTS TRIES TO GO NEGATIVE GO
           TO FIBONACCI
C*****SEARCH
      IF(KEY.EQ.1)GO TO 100
    7 IF(FC.GT.FB)GO TO 100
      ALFM2=ALFM1
      ALFM1=ALFK
      ALFK=ALFK*(1.+R**(K-1)
   10 CONTINUE
      WRITE(3,11)
   11 FORMAT (T20,'NO ALPHA MIN FOUND')
      ALF=1.
      RETURN
  100 CALL FIBMIN(ALFM2,ALFK,ALF,8,F,FA,FC,S)
      RETURN
      END
C
      SUBROUTINE FIBMIN(X1,X2,XMIN,ITER,F,F1,F2S)
C*****THIS SUBROUTINE MINIMIZES A FUNCTION F BY FIBONACCI
           SEARCH
      IMPLICIT REAL*8(a=H,0-Z)
      DIMENSION S(12),X(12),DESIRE(30),WDES(30),XSWAP(12)
      COMMONX,DESIRE,WDES,XSWAP,NEL,NW
  100 IY1=1
```

```
      IY2=1
    2 J=0
C*****CALCULATE APPROPRIATE FIBONACCI NUMBERS
   30 DO 3 I=1,ITER
      ITEMP=IY2
      IY2=IY2+IY1
    3 IY1=ITEMP
      Y1=IY1
      Y2=IY2
C*****DETERMINE FIRST SEARCH POINT
    1 XA=X1+(X2-X1)*Y1/(Y1+Y2)
      GO TO 400
C*****DETERMINE NEXT SEARCH POINT AND VALUE OF FUNCTION
  300 IF(KEY)14,14,13
   13 FA=FX(XA,S,KEY1)
      IF(KEY1-1) 6,7,6
    7 X2=XA
      F2=FA
      GO TO 100
   14 FB=FX(XB,S,KEY1)
      IF(KEY1-1)6,15,6
   15 F2=FB
      X2=XB
      GO TO 100
    6 J=J+1
C*****DETERMINE MINIMAL SEARCH POINT
    8 IF(FA-FM)4,4,5
    4 X2=XB
      F2=FB
      KEY=1
      XB=XA
      FB=FA
C*****DETERMINE NEXT SEARCH POINT AND VALUE OF FUNCTION
      XA=X1+X2-XB
  400 XMIN=XA
   21 IF(J-ITER)200,9,9
  200 FA=FX(XA,S,KEY1)
      IF(KEY1-1)500,7,500
    5 X1=XA
      F1=FA
      XA=XB
      FA=FB
  500 KEY=-1
      XB=X2-XA+X1
      XMIN=XB
      GO TO 300
C*****RETURN APPROPRIATE VALUE FOR XMIN
 1000 FORMAT(T10,110(1H-),/,T20,'ALPHAMIN NOT EXACT')
    9 IF(KEY)10,10,11
   10 XMIN=XA
      F=FA
      IF(KEY1.EQ.1)WRITE(3,1000)
```

```
      RETURN
   11 XMIN=XB
      F=FB
      IF(KEY1.EQ.1)WRITE(3,1000)
      RETURN
      END
C
      DOUBLE PRECISION FUNCTION FX(ALF,S,KEY)
C*****THIS FUNCTION FINDS F(X+ALF*S)
C*****IF KEY = 1 ONE OF THE ELEMENTS IS NEGATIVE
      IMPLICIT REAL*8(A-H,0-Z)
      DIMENSION X(12),DESIRE(30),WDES(30),XSWAP(12),S(12)
      COMMON X,DESIRE,WDES,XSWAP,NEL,NW
      KEY=0
      DO 10 I=1,NEL
      XSWAP(I)=X(I)
      X(I)=X(I)+ALF*S(I)
      IF(X(I))5,10,10
    5 KEY=1
C*****WRITE WHICH ELEMENT TENDS NEGATIVE
      WRITE(3,1) I
    1 FORMAT(T49,'ELEMENT NO.',I2,' TENDS TO NEGATIVE')
   10 CONTINUE
      FX=ERRF(1)
      DO 20 I=1,NEL
   20 X(I)=XSWAP(I)
      RETURN
      END
C
      DOUBLE PRECISION FUNCTION ERRF(L)
C*****THIS FUNCTION EVALUATES THE ERROR FUNCTIONAL
      IMPLICIT REAL*8(A-H,0-Z)
      DIMENSION DESIRE (30,WDES(30,V(10),X(12),PST(8),
           XSWAP(12)
      COMMON X,DESIRE,WDES,XSWAP,NEL,NW
      COMPLEX*16 V,PSI
      L=1
      ERRF=0.D0
      DO 10 I=1,NW
      W=WDES(I)
      CALL NETWK(V,W,1,PSI)
      VAL=CDABS(V(7))-DESIRE(I)
      ERRF=ERRF+VAL*VAL
   10 CONTINUE
      RETURN
      END
C
      SUBROUTINE NETWK(V,W,KEY,PSI)
C*****THIS SUBROUTINE SOLVES THE NETWORK MODEL FOR THE
           OUTPUT VOLTAGE
C*****OR ALL VOLTAGES IF KEY IS NOT EQUAL TO 1
C*****IF KEY EQUALS 3 THE ADJOINT VOLTAGES ARE CALCULATED
```

```
      IMPLICIT REAL*8(A-H,0-Z)
      DIMENSION Y(8,8),V(10),X(12),PSI(8),DESIRE(30),
           WDES(30),XSWAP(12)
      COMPLEX*16 PIVOT,Y,V,DCMPLX,PSI,DCONJG,PSIX
      COMMON X,DESIRE,WDES,XSWAP,NEL,NW
C
C*****INITIALIZATION
C
      KEY1=2
C*****PSI(8) CONTAINS THE DESIRED OUTPUT VOLTAGE IF
           ADJOINT METHOD USED
      IF(KEY.NE.3) GO TO 2
      PSIX=PSI(8)
    2 G2=1./X(1)
      G3=1./X(2)
      G4=1./X(3)
      G5=1./X(4)
      G6=1./X(5)
      G7=1./X(6)
      G8=1./X(7)
      G9=1./X(8)
      G10=1./X(9)
      C3=X(10)
      C4=X(11)
      C5=X(12)
    3 DO 7 L=1,8
      IF(KEY1.EQ.3) GO TO 6
      V(L)=(0.D0,0.D0)
    6 PSI(L)=(0.D0,0.D0)
      DO 7 M=1,8
      Y(L,M)=(0.D0,0.D0)
    7 CONTINUE
C*****THESE ELEMENTS ARE COMMON TO BOTH THE INITIAL CIRCUIT
           AND ADJOINT NETWORK ADMITTANCE MATRICES
      Y(2,2)=DCMPLX(1.2+G7,W*C3)
      Y(2,3)=DCMPLX(-G7,0.D0)
      Y(2,4)=DCMPLX(0.D0,-W*C3)
      Y(3,2)=Y(2,3)
      Y(3,3)=DCMPLX(G7+G8,W*C5)
      Y(3,7)=DCMPLX(-G8,0.D0)
      Y(4,2)=Y(2,4)
      Y(4,4)=DCMPLX(G9,W*(C4+C3)
      Y(4,7)=DCMPLX(0.D0,-W*C4)
      Y(5,5)=DCMPLX(321.55+G5,0.D0)
      Y(6,6)=DCMPLX(382.04+G6,0.D0)
      Y(7,3)=Y(3,7)
      Y(7,4)=Y(4,7)
      Y(7,7)=DCMPLX(.04+G8+G10,W*C4)
C*****THESE ELEMENTS ARE IN THE ADJOINT MATRIX
      IF(KEY1.NE.3)GO TO 1
      Y(1,1)=DCMPLX(11.64+G4,0.D0)
      Y(1,5)=(-9.64D0,0.D0)
      Y(1,6)=(-392.D0,0.D0)
```

```
      Y(1,7)=(392.D0.0D0)
      Y(2,5)=(-313.D0,0.D0)
      Y(5,1)=(-9.56D0,0.D0)
      Y(5,2)=(-1.2D0,0.D0)
      Y(6,1)=(-2.D0,0.D0)
      Y(6,6)=DCMPLX(382.04+G6,0.D0)
      Y(6,7)=(-380.04D0,0.D0)
      Y(7,6)=(-.04D0,0.D0)
      PSIX=PSIX*V(7)/CDABS(V(7))
      PSI(7)=DCONJG(V(7))-DCONJG(PSIX)
      GO TO 4
C*****THESE ELEMENTS ARE IN THE ORIGINAL MATRIX
    1 Y(1,1)=DCMPLX(2.04+G4,0.D0)
      Y(1,5)=(-9.56D0,0.D0)
      Y(1,6)=(-2.D0,0.D0)
      Y(1,8)=(-2.3D0,0.D0)
      Y(2,5)=(-1.2D0,0.D0)
      Y(5,1)=(-.04D0,0.D0)
      Y(5,2)=(-312.D0,0.D0)
      Y(5,8)=(2.31D0,0.D0)
      Y(6,1)=(-382.D0,0.D0)
      Y(6,7)=(-.04D0,0.D0)
      Y(7,1)=(380.D0,0.D0)
      Y(7,6)=(-380.04D0,0.D0)
C
C*****MATRIX REDUCTION
C
    4 PIVOT=Y(1,1)
      D0 10 I=5,8
   10 Y(1,I)=Y(1,I)/PIVOT
      Y(5,8)=Y(5,8)-Y(5,1)*Y(1,8)
      IF(KEY1.EQ.3)Y(5,7)=-Y(5,1)*Y(1,7)
      Y(5,6)=-Y(1,6)*Y(5,1)
      Y(6,8)=-Y(6,1)*Y(1,8)
      IF(KEY1.EQ.3)Y(6,7)=Y(6,7)-Y(6,1)*Y(1,7)
      Y(6,6)=Y(6,6)-Y(6,1)*Y(1,6)
      Y(6,5)=-Y(6,1)*Y(1,5)
      IF(KEY1.EQ.3)GO TO 12
      Y(7,8)=-Y(7,1)*Y(1,8)
      Y(7,6)=Y(7,6)-Y(7,1)*Y(1,6)
      Y(7,5)=-Y(7,1)*Y(1,5)
   12 PIVOT=Y(2,2)
      D0 20 I=3,5
   20 Y(2,I)=Y(2,I)/PIVOT
      Y(3,5)=-Y(3,2)*Y(2,5)
      Y(3,4)=-Y(3,2)*Y(2,4)
      Y(3,3)=Y(3,3)-Y(3,2)*Y(2,3)
      Y(4,5)=-Y(4,2)*Y(2,5)
      Y(4,4)=Y(4,4)-Y(4,2)*Y(2,4)
      Y(4,3)=-Y(4,2)*Y(2,3)
      Y(5,5)=Y(5,5)-Y(5,2)*Y(2,5)
```

```
      Y(5,4)=-Y(5,2)*Y(2,4)
      Y(5,3)=-Y(5,2)*Y(2,3)
      PIVOT=Y(3,3)
      D0 30 I=4,7
   30 Y(3,I)=Y(3,I)/PIVOT
      Y(4,7)=Y(4,7)-Y(4,3)*Y(3,7)
      Y(4,5)=Y(4,5)-Y(4,3)*Y(3,5)
      Y(4,4)=Y(4,4)-Y(4,3)*Y(3,4)
      Y(5,7)=-Y(5,3)*Y(3,7)
      Y(5,5)=Y(5,5)-Y(5,3)*Y(3,5)
      Y(5,4)=Y(5,4)=Y(5,3)*Y(3,4)
      Y(7,7)=Y(7,7)-Y(7,3)*Y(3,7)
      Y(7,5)=Y(7,5)-Y(7,3)*Y(3,5)
      Y(7,4)=Y(7,4)-Y(7,3)*Y(3,4)
      Y(4,5)=Y(4,5)/Y(4,4)
      Y(4,7)=Y(4,7)/Y(4,4)
      Y(5,7)=Y(5,7)-Y(5,4)*Y(4,7)
      Y(5,5)=Y(5,5)-Y(5,4)*Y(4,5)
      Y(7,7)=Y(7,7)-Y(7,4)*Y(4,7)
      Y(7,5)=Y(7,5)-Y(7,4)*Y(4,5)
      D0 40 I=6,8
   40 Y(5,I)=Y(5,I)/Y(5,5)
      Y(6,8)=Y(6,8)-Y(6,5)*Y(5,8)
      Y(6,7)=Y(6,7)-Y(6,5)*Y(5,7)
      Y(6,6)=Y(6,6)-Y(6,5)*Y(5,6)
      Y(7,8)=Y(7,8)-Y(&,5)*Y(5,8)
      Y(7,7)=Y(7,7)-Y(7,5)*Y(5,7)
      Y(7,6)+Y(7,6)-Y(7,5)*Y(5,6)
      Y(6,7)=Y(6,7)/Y(6,6)
      Y(6,8)=Y(6,8)/Y(6,6)
      Y(7,8)=Y(7,8)-Y(7,6)*Y(6,8)
      Y(7,7)=Y(7,7)-Y(7,6)*Y(6,7)
      Y(7,8)=Y(7,8)/Y(7,7)
C*****CALCULATE ADJOINT VOLTAGES
      IF(KEYl.NE.3)GO TO 9
      PSI(7)=PSI(7)/Y(7,7)
      PSI(6)=-Y(6,7)*PSI(7)
      PSI(5)=-Y(5,7)*PSI(7)-Y(5,6)*PSI(6)
      PSI(4)=-Y(4,7)*PSI(7)-Y(4,5)*PSI(5)
      PSI(3)=-Y(3,7)*PSI(7)-Y(3,5)*PSI(5)-Y(3,4)*PSI(4)
      PSI(2)=-Y(2,5)*PSI(5)-Y(2,4)*PSI(4)-Y(2,3)*PSI(3)
      PSI(1)=-Y(1,7)*PSI(7)-Y(1,6)*PSI(6)-Y(1,5)*PSI(5)
      RETURN
C*****CALCULATE NOTE VOLTAGES BY BACK SUBSTITUTION
    9 V(7)=Y(7,8)
C*****IF KEY EQUALS 1 ONLY THE OUTPUT VOLTAGE IS DESIRED
      IF(KEY.EQ.1)GO TO 50
      V(6)=Y(6,8)-Y(6,7)*V(7)
      V(5)=Y(5,8)-Y(5,7)*V(7)-Y(5,6)*V(6)
      V(4)=-Y(4,5)*V(5)-Y(4,7)*V(7)
      V(3)=-Y(3,4)*V(4)-Y(3,5)8V(5)-Y(3,7)*V(7)
      V(2)=-Y(2,3)*V(3)-Y(2,4)*V(4)-Y(2,5)*V(5)
```

```
      V(1)=Y(1,8)-Y(1,5)*V(5)-Y(1,6)*V(6)
      KEY1=KEY
      IF(KEY1.EQ.3)GO TO 3
   50 RETURN
      END
```